Learning Journal

Student Name: Jinish Vaidya

Course: Software Project Management

Journal URL: https://github.com/Jinish-Vaidya/Software-Project-Management/tree/main/Learning_Journal

Week 1: 20th January to 24th January, 2023

Date: 24th January, 2023

Key Concepts Learned:

- Growth of IT and software sector in present and future,
- Job vs project vs exploration
  - Job is repetition of task with very less uncertainty while project.
  - Project lies between job and exploration where there is particular start and end time in which one have to achieve some predefined goals.
  - When there is no guarantee of output then it is called exploration. Exploration is highly uncertain.

- What is project management?
  - Management of project with limited amount resources, budget, and time.
  - Phases involved- project initiation, project planning, project monitoring and control, and project closure.
- What is software project management?
  - Software project management = Project management + Software engineering
  - With different phases it also include requirement development, software design, software construction, software testing, and software maintenance in project planning and project monitoring phase.

- Difference of software project wrt project?
  - Invisibility
  - Complexity
  - Conformity
  - Flexibility

- IT and software difference- IT includes software system, hardware system, and other then it.

- Software development + software maintenance = software project.

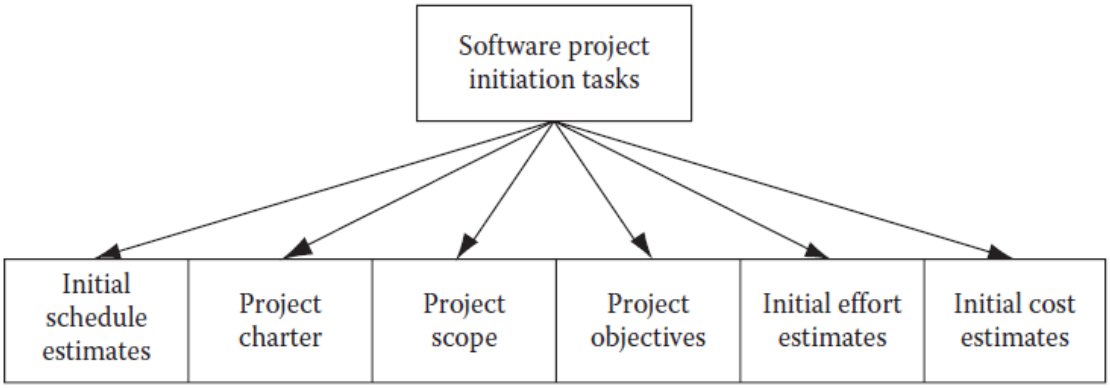
- Software application vs software product

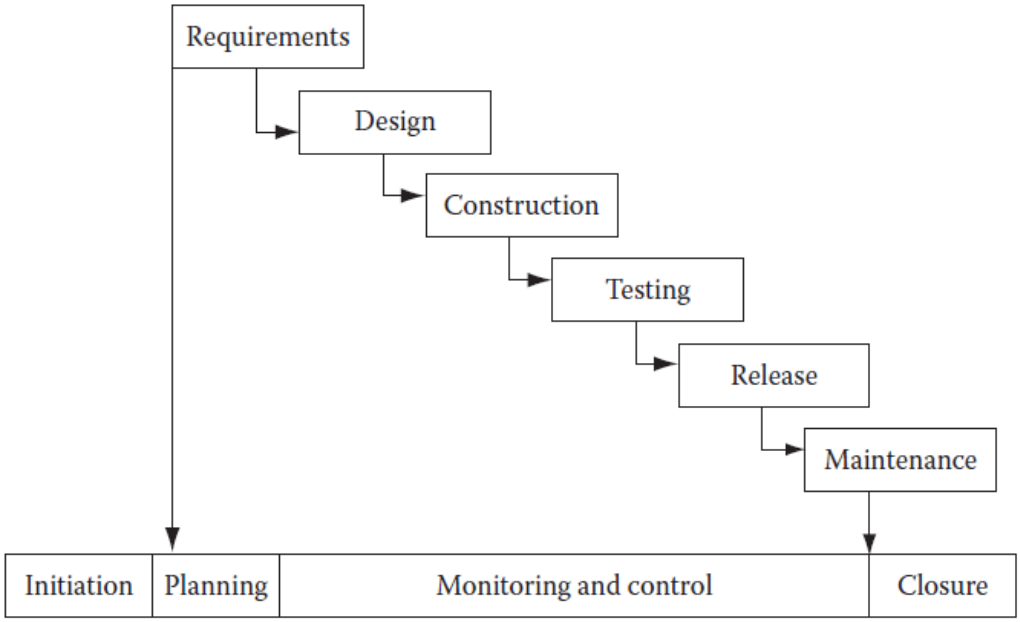
- When software is developed for the use of organization itself then it is called software application.
- When software is developed for the purpose of selling to customer and for the organization use then it is called software product. And the organization that develop it is called software vendors.

- Project management process
  - There are three kind of process running in an organization to develop software process and application that is software life cycle processes, project management processes, and organization level processes.

- How are these software products made? After all, development of these software products does not start with end-user requirements. ¡e software vendor sees a market opportunity of developing such a product.

- software application is created based on end-user requirements, a software product is made using market research data.



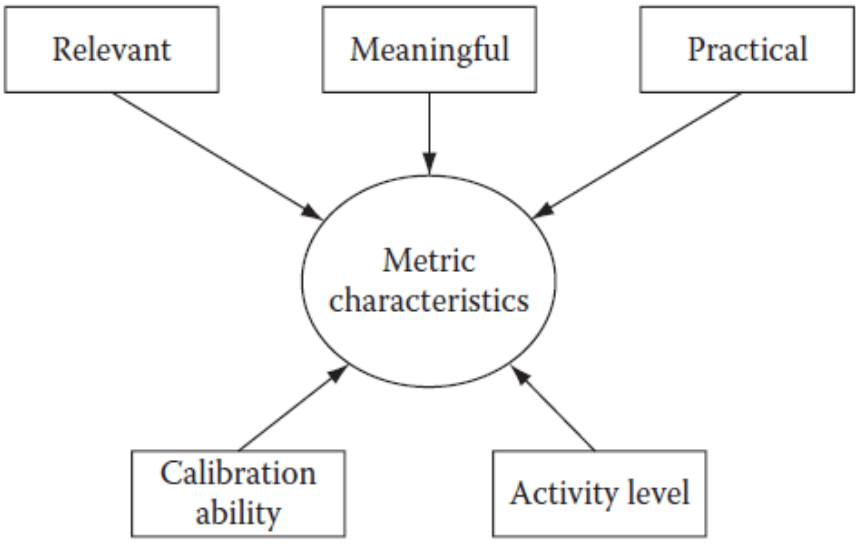


- Configuration and version control management
  - Change in requirement are encountered during project development life cycle. Due to this word done in project development life cycle also need changes. Due to which many versions are produced during all phases of project development life cycle. Managing all these work products is done using configuration and version control.

- The best solution for managing various requirement versions is to have a central repository where all versions of requirements can be stored.

- Management Metric

- In the case of software development projects, the management metrics are the productivity data for the projects.



- Measurement should be done at minute level and not at gross level. Gross level measurements fail to point to the root causes of problems.

- Many of these approaches use statistical process control (SPC) methods.
- SPC approach is popularly known as the Seven Tools of Quality as it uses 7 distinct techniques.

- Different role of project manage of inhouse and outsource project for initial hiccups and false starts due to unclear project charter, an unclear project scope and unclear requirements.

Project charter

Project charter is made by the top management of the organization for starting a software project. Project charter basically defines the purpose for starting the project.

Project scope

Project scope is developed to define boundaries of the project. The scope will include what functionalities are needed in the software product to be developed. It will also define level of quality needed in the software product.

Project objective

The stakeholders state and set the project objectives. ¡e objectives should be stated in clear language
and the set of objectives should be kept as small as possible.

Some of the factors that make project management vary for different projects are as follows:
Project size, Product quality, Technology, Code reuse.

Estimate Initial Project Size

Why Initial project size is needed?
Rough project size should be estimated so that a sketch of the initial project plan can be realized.

Estimate Initial Project Effort and Costs

The initial project cost estimates are determined based on team productivity, effort estimates, hours contributed by software professionals, and their hourly rates. Stakeholders prioritize cost considerations, and if a project exceeds budget expectations, it may be reconsidered or scaled down. Using data from Figures 2.1 through 2.3, a project with a 14-month schedule and 56 man-months effort is estimated. With an average team member salary of $4000/month and 15% overhead costs, the tentative development cost is $268,800. Early cost estimation is crucial for customer satisfaction and project initiation.

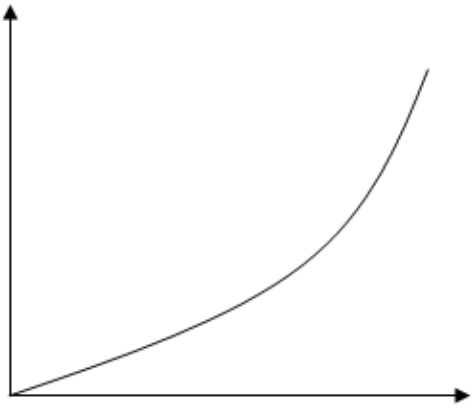
Project schedule

Project schedule is vital for gaining a competitive edge, with time-sensitive objectives driving stakeholders to push for timely software implementation. During initiation, stakeholders may prioritize an accelerated schedule, even if it leads to increased costs. In such cases, the project manager must adjust the plan and resource allocation to meet stakeholders' timeline requirements.

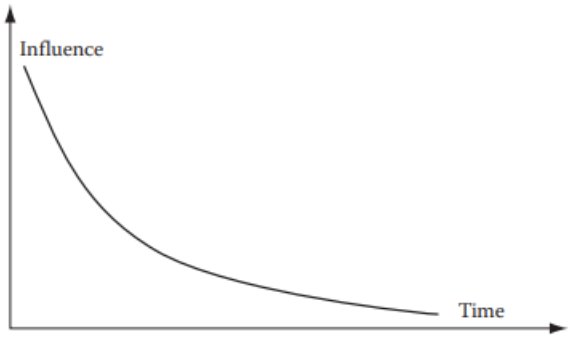
Tentative Project Plan

- **Project schedule**
- **Project cost**
- **Communication Plan**
- **Resource Plan**
- **Quality Plan**
- **Tool Plan**
- **Risk Plan**

Tentative Project schedule



Stakeholder Influence graph



Application in Real Projects:

- Third-party logistics service providers (3PL) to get instant information about the need to have trucks for transportation of goods by its customers.
- It can be used in a very sophisticated appointment scheduling of trucks at both receiving and shipping warehouses.
- Project charter, Project scope, and project object should be clearly define so that initial hiccups and false starts can be prevented up to greater extent.

Benefits

- The warehouse staff just has to execute as per available details.
- There will be no loss of time anywhere right from truck arrangement for loading to unloading of truck.

Peer Interactions:
Interaction about case study and software project management concepts, its importance and role of project manage.

Goals for the Next Week:
Go through chapter 3,4 and 5

Week 2: 28th January – 3rd February

Date: 3rd February

Key Concepts Learned:

- Effort estimation techniques like Function Point Analysis (FPA), COCOMO, and Wide Band Delphi are crucial for project planning and resource allocation in software development projects.

- Resource allocation and loading factors play a vital role in optimizing workforce capacity for efficient task handling.

- Continuous product development in software projects emphasizes the iterative nature of development, requiring ongoing effort and cost estimation for long-term project success.

- A case study of a Software as a Service (SaaS) vendor provided practical insights into estimating project size, incremental software development, and continuous operation.

- Introduction of new terms and methodologies such as loading factor, continuous product development, and SLOC (Source Lines of Code) enhanced understanding of effort estimation and project management in software development.

- Started with project and made survey for market analysis.

Application in Real Projects:

- **Resource Allocation and Budgeting**: Effort and cost estimations guide the allocation of resources such as personnel and equipment, as well as budgeting for various project activities, ensuring efficient resource utilization and financial planning.
- **Project Planning and Risk Management**: Estimations aid in creating realistic project schedules, identifying potential risks, and developing mitigation strategies, enabling proactive planning and management of project timelines and potential challenges.
- **Performance Measurement and Continuous Improvement**: Estimations serve as benchmarks for measuring project performance, facilitating analysis of deviations and informing continuous improvement efforts to refine estimation techniques and enhance future project outcomes.

Collaborative Learning:
During collaborative learning, we discussed the market analysis phase and collaboratively created a market survey form to facilitate our market analysis efforts.

Challenges Faced:
Understanding effort estimation techniques such as Function Point Analysis, COCOMO, and Wide Band Delphi, as well as comprehending concepts related to resource allocation and loading factors in optimizing workforce capacity for efficient task handling. These areas required further clarification and additional effort to grasp fully and apply effectively in project planning and management.

Reflections on Case Study
The case study of the SaaS vendor shows how important it is to estimate the effort needed for a project. This helps with planning the project, assigning resources, and managing finances in real-world software development. The vendor initially estimated they'd need 500,000 lines of code (SLOC) and used incremental development, which allowed them to adapt to changes in the market and use their resources efficiently. They regularly checked their costs to make sure they could keep going financially, and they focused on making continuous improvements to ensure the project's success. Overall, the case study highlights how crucial it is to estimate effort accurately to make good decisions and improve project outcomes in the ever-changing world of software development.

Personal development activities:
Engaged in learning sessions focused on effort estimation techniques like FPA, COCOMO, and Wide Band Delphi to enhance my project planning skills.

Adjustments to Goals:
Complete 1st and 2nd chapter in 1st week and gone through 3rd chapter in 2nd week. Also done market analysis through survey form for project.

Goals for the Next Week:
Reading risk management chapter and going to work on project initiation.

Week 3: 4th February – 10th February

Date: 10th February

Key Concepts Learned:
Types of Risks in Software Projects: There are various types of risks that can impact software projects, including external and internal risks. External risks have factors such as market changes, technology obsolescence, and vendor reliability, while internal risks involve budget constraints, schedule overruns, and quality compromises. The concept of balancing product quality, project budget, and schedule is highlighted as a critical consideration in risk management.

Risk Mitigation Strategies: The chapter emphasizes the importance of clear communication and setting limits for the project team to ensure that they understand and deliver within these boundaries. Additionally, the project manager is advised to plan effectively to handle unexpected situations and surprises that may arise during the project.

Quality Planning and Review Processes: Quality risks are identified as a significant concern in software projects, and the chapter introduces the concept of integrating quality checks into the project schedule to ensure that work products meet the desired level of quality. It also advocates for peer reviews, code reviews, and formal quality review processes to maintain overall product quality.

Technology Risks and Mitigation: The chapter highlights the risk of technology obsolescence and the potential impact on software products. It introduces the concept of selecting appropriate programming languages, hardware platforms, and user access methods to prevent software products from becoming obsolete. The importance of engaging with vendors to ensure future support for technology tools and techniques is also emphasized.

Offshore Project Management Strategies: The chapter presents strategies for mitigating risks associated with offshore software development, including competency and maturity checks of offshore service providers, drafting comprehensive service level agreements (SLAs), addressing attrition through employee counseling and performance reviews, and implementing thorough checks for tasks performed by offshore teams.

New Terms, Methodologies, and Frameworks:
Service Level Agreement (SLA): A contract between a service provider and a customer that defines the level of service expected from the service provider.
Offshore Service Providers: Companies or teams located in a different country or geographical location, often engaged in software development or other IT-related services.
Quality Planning: The process of integrating quality checks and review processes into the project schedule to ensure the desired level of quality in work products.
Technology Obsolescence: The risk associated with technology becoming outdated or obsolete, potentially rendering software products unusable.

The chapter provides valuable insights into the complexities of software project risk management, offering practical strategies and considerations for mitigating various types of risks. It underscores the critical role of effective communication, clear requirements, and proactive planning in addressing potential challenges throughout the project lifecycle.

Application in Professional Life:

Project Planning & Execution:

Understand market changes, budget constraints, technology trends to plan effectively.
Set clear boundaries for your team to ensure smooth execution within the plan.

Quality Assurance:

Integrate quality checks throughout the project (peer reviews, code reviews) to maintain desired quality.
Crucial for projects with strict quality standards or regulations.

Vendor Management:

Understand vendor reliability risks and draft clear Service Level Agreements (SLAs).
Proactive engagement with vendors ensures alignment with project goals.

Technology Selection & Management:

Choose future-proof technology (languages, platforms) and secure vendor support to avoid obsolescence.
Makes software products sustainable and adaptable to future updates.

Offshore Project Management:

Address competency, attrition, cultural differences risks in offshore teams.
Implement competency checks, thorough SLAs, and performance reviews for mitigation.

Communication & Stakeholder Management:

Communicate risks and mitigation strategies transparently to stakeholders.
Build trust and ensure everyone is aligned towards project objectives.

Overall:

Knowledge of software project risk management helps plan, execute, and mitigate risks better.
Promotes proactive decision-making and fosters a risk-aware culture essential for project success.

Further Applications:

Applicable beyond software projects - construction, finance, healthcare, etc.
Helps manage any project with potential risks and ensure smoother execution.

Reflections on Case Study

The case study presented in this chapter is on challenges and risks faced by a SaaS vendor during the development of its flagship software product.

The identified risks, such as the viability of offshore teams, attrition, communication gaps, development costs, schedule management, and software product quality, underscore the multifaceted nature of risks in software projects.

The case study highlights the importance of conducting thorough risk assessments and formulating mitigation plans to address potential challenges proactively.

It showcases the critical role of risk management in ensuring the success of software projects and the need for strategies to mitigate risks effectively.

The emphasis on market research, strategy formulation, risk assessment, and risk mitigation in the case study underscores the holistic approach required to navigate uncertainties in software development.

The case study serves as a practical example of how businesses can identify, analyze, and manage risks to enhance project outcomes and achieve their strategic objectives.

Overall, the case study reinforces the significance of risk management in software projects and the value of proactive planning and mitigation strategies in mitigating potential risks and ensuring project success.

Collaborative Learning:
We discussed and made analysis about the market in real world by answer we got from the survey on our topic Collaborative Project Management for Creative Teams. In which various question were answered by many Corporate Professionals and few students from various countries. The survey aimed to identify the project management tools regularly used by participants and to gather insights into their pain points and desired features in such tools.

Target audience: The primary target audience is project managers and other professionals involved in project management, across various industries and company sizes.

Market size: The market for collaborative project management tools is large and growing, with a CAGR of 10.67% from 2024 to 2029. This growth is driven by the increasing complexity of projects, the shift towards remote work, and the need for more efficient and effective project management tools.

Competitors: The main competitors include GitHub, Jia, Microsoft Project, Canva, Google Docs, and Azure Devops. Each competitor has its own strengths and weaknesses.

Pain points: Lack of predefined templates, absence of diagrams for status updates, outdated formats, lengthy loading times, steep learning curves, and issues with permissions management.

Customer needs: Customers are looking for project management tools that are easy to use, offer real-time collaboration features, integrate with other tools, and provide robust reporting and analytics.

Adjustments to Goals:

- Read risk management chapter and its case study.
- completed its Exercise 1st question.
- Gone through project initiation phase, contributed in report preparation and analyse the responses from survey obtained based on topic Collaborative Project Management for Creative Teams.

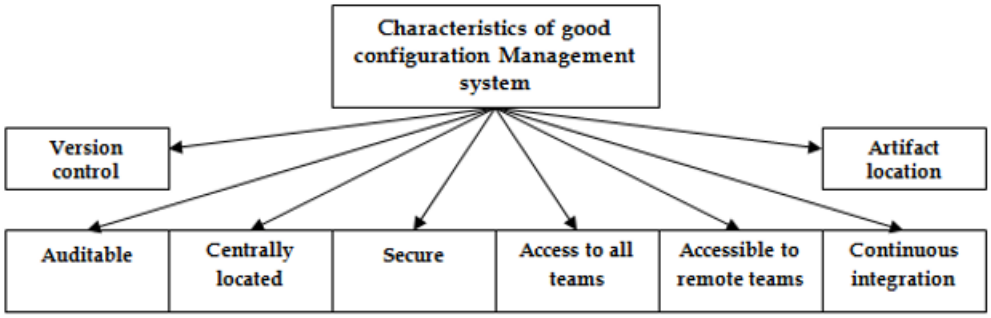
Goals for next week

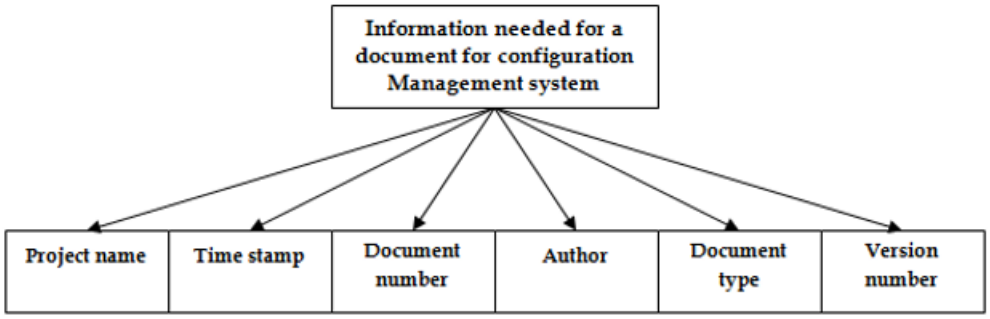
- Start with feasibility study part of the project study to assess the technical, operational, and economic viability.
- Going to read configuration management chapter.
- We are going to make PowerPoint presentation and prepare for project pitch presentation.
- Meet and have discussion with another team member regarding project pitch next week.

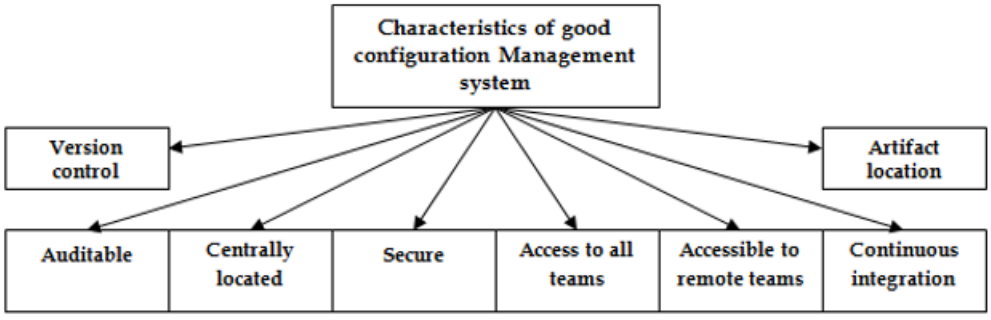
Week 4: 11th February – 17th February

Date: 17th February

Key Concepts Learned:
Configuration Management System (CMS): The importance of a centralized CMS for managing software development projects efficiently. It includes version control, access control, branching mechanisms, and audit facilities to ensure that team members work on the correct versions of documents and artifacts.







Incremental Iteration Development Model: Case study of a software vendor using the incremental iteration development model. This model involves developing software in incremental stages, with a focus on continuous integration and efficient configuration management across internal and offshore teams.

Best Practices in Configuration Management: In this chapter I learnt about best practices for configuration management systems, such as centralized systems, role-based access control, continuous integration with smoke testing, easy branching mechanisms, and audit facilities. These practices help in maintaining the integrity of software builds and managing versions effectively.

Artifact Management: Configuration management systems store various artifacts generated during the software development lifecycle, including requirement specifications, design documents, software builds, testing plans, and training manuals. These artifacts undergo versioning to track changes and ensure traceability.

Decentralized Configuration Management: The document contrasts centralized and decentralized configuration management systems, highlighting the challenges of synchronizing versions across multiple systems. It stresses the importance of a centralized approach for smoother functioning and reduced overhead in managing software projects.

By understanding these concepts and implementing the recommended practices, project teams can streamline their development processes, ensure version control, facilitate collaboration among distributed teams, and maintain the integrity of software builds throughout the project lifecycle.

Reflections on Case Study

Case study is presented regarding a U.S.-based mid-market software vendor that developed a software system for managing orders, inventories, and logistics services. The company adopted an incremental iteration development model and utilized both internal project teams and offshore service providers in locations like India and Russia to reduce costs and accelerate development cycles.

Key points from the case study include:

The company successfully implemented a centralized configuration management system accessible to all teams, regardless of their locations.
The configuration management system operated 24/7 with high security measures in place, ensuring minimal downtime and no security breaches.
Access rights were managed effectively, with different levels of permissions granted to team members based on their roles and responsibilities.
The main branch of the version control system contained the primary software build with all major updates since the product's development, along with related artifacts.
The workflow included source code check-ins, automatic smoke testing after code compilation, and notifications for test results, ensuring the reliability and quality of the software build.
This case study highlights the importance of efficient configuration management in supporting distributed development teams, ensuring version control, and maintaining the security and integrity of software projects throughout their lifecycle.

Collaborative Learning:

- Pitch Presentation Preparation: I worked with my team to create visually appealing slides for our pitch presentation. We carefully crafted the content, organized our ideas into a coherent structure, and paid attention to the design elements to enhance visual appeal. Additionally, we discussed various aspects of how to deliver the pitch effectively, and emphasis on key points.

- File Sharing Platform: We set up a common file-sharing platform to track our project's progress. This platform helps us organize our files, maintain version control, assign tasks, and collaborate seamlessly.

Adjustments to Goals:

- Read configuration management chapter and its case study.
- Gone through project initiation phase, contributed in report preparation and prepared the power point presentation for pitch and discussed about it.

<u>Goals for next week</u>

- Going to read project planning chapter.
- Going to refer Introduction to Software Project Management, Project Initiation Management, Software Project Effort and Cost Estimation, Risk Management, Configuration Management for mid term 1 preparation.