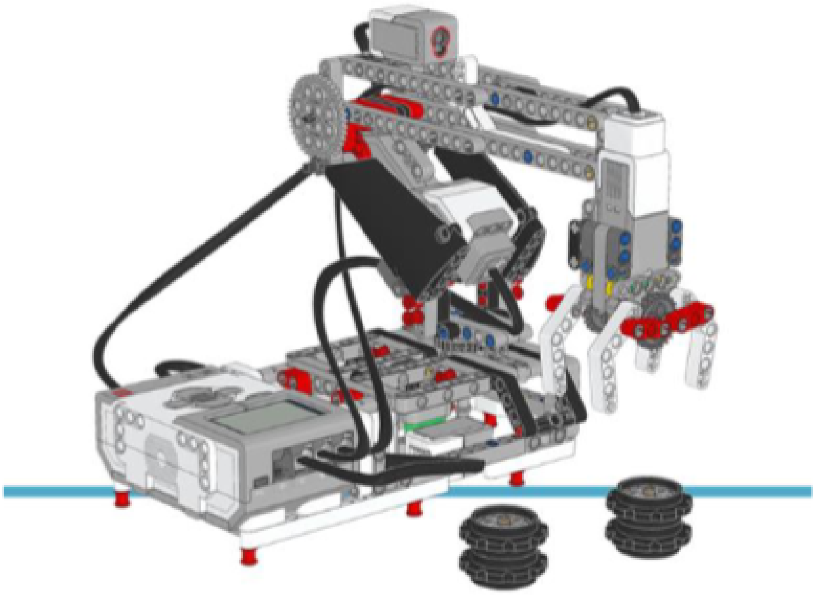
Project Plan for Robot Package Handling System



Rasmus Eliasson, Omar Idris, Subhi Alloush,
Mohammad Aljubran, Mohamad Al darwich
Group 7

March 7, 2024

Contents

1 Introduction

1.1 Project overview

In a warehouse, one of the primary tasks is to categorize incoming packages according to their types. This process can pose a risk of injury to the workers, given the unusual shapes and additional weight of the packages.

1.2 Objectives & goals

Utilize robots for sorting packages in a warehouse based on various properties, such as colour, shape, and content. The robots should be designed to identify, recognize, pick up, and respectively place down the packages.

The environment in which the robots operate has been delineated with lines to indicate areas where the robots can function without posing a risk of injury to humans.

2 Project Organization

2.1 Memo - Communication

During the kick-off meeting, we inquired with the product owner about the product's functionalities. The product owner emphasized the need for careful consideration of sensors and the selection of appropriate motors. Additionally, he advised us to revise and further refine the existing requirements for enhanced clarity and ease of implementation. Furthermore, the product owner engaged in a discussion about risk management with our team. We also recieved news of late changing in requirments in the second workshop.

2.2 Contribution team members

All team members collaborted on brainstorming robot functionallity and design. See bullet below for more specific task participation.

Muhammed Al Darwich

- Refine requirments
- WBS
- Risk Managment
- Inspecting robot

Rasmus

- Refine requirements
- Critical path
- Came up with robot design (which was later scratched)
- Inspected hardware ports
- Inspected the rotation gears
- Helped everyone write in LaTeX
- Engaged in writing code for rotation movement
- Engaged in writing code for simple arm movement
- Helped with estimation, GANTT
- Setting milestones for demos
- Wrote introduction page
- Risk Management

Muhammed Aljubran

- Refine requirements
- Research pybricks documentation
- Inspected the rotation gears
- Inspected the arm elevation gears
- Engaged in assisting writing code for rotation movement
- Engaged in assisting writing code for simple arm movement
- Risk management
- Flow chart

Omar

- Refine requirements
- Critical path
- GANTT
- WBS & WBS figure
- Testing strategies
- Setting milestones
- Meeting documentation
- Risk Management

Subhi

- Refine requirements
- GANTT
- WBS briefing
- WBS
- Resource allocation
- Risk Management
- Memo

3 Risk Management

3.1 Risk Identification

Risk Identification	
Risk Type	Possible Risks
People	Ill staff (Sick leave), Miscommunication(between staff also between customer)
Estimation	Delayed development, incorrect calculation of time, cost and resources.
Organizational	Financial Problems
Requirements	Customer complaints
Technology	Power failure, Data Security (for example getting hacked), bad environment when testing
Tools	Code generation tools are inefficient, with insufficient hardware and insufficient tools for maintenance

In the context of the project development, risk identification is crucial for anticipating and mitigating potential challenges that may arise during the project development. Below, we have listed various types of risks categorized based on their nature:

People:

- **I'll Staff (Sick Leave):** One potential risk involves staff members falling ill or requiring extended absence for various reasons. Which could impact resource availability and also consequently the project's development timeline.
- **Miscommunication:** Another risk within the people category is miscommunication, both among project team members and between the team and the customer. This can lead to misunderstandings, delays, and potential rework.

Estimation:

- **Delayed Development:** Risk of delayed development arises from inaccurate estimation of project tasks, task order dependencies, and appropriate resource allocation. This could result in project delays and budget overruns.
- **Incorrect Calculation of Time, cost and Resources:** Inaccurate estimation of project parameters such as time, cost and resource requirements poses a significant risk to the projects success.

Organizational:

- **Financial Problems:** Within the category of organizational risks, includes financial challenges. Challenges such as unexpected expenses, budget constraints, or funding issues that could impact the project's progress and viability.

Requirements:

- **Customer Complaints:** Risks associated with requirements involve dissatisfaction or complaints from the customer regarding the delivered product or service not meeting their expectations or needs. As well as suddenly adding new requirements, when the development already is underway, resulting in potential extra rework.

Technology:

- **Power Failure:** Technical risks encompass potential disruptions due to power failures, which could lead to data loss, system downtime, and project delays.
- **Data Security:** Risks related to data security involve the potential for unauthorized access, data breaches, or cyber-attacks compromising sensitive project information or sabotage of the development.
- **Bad Environment When Testing:** Technical risks also include challenges related to testing environments, such as inadequate facilities, insufficient equipment, or environmental factors impacting testing outcomes. Could be lights affecting the sensors or other disturbances.

Tools:

- **Code Generation Tools are Inefficient:** Risks associated with tools include inefficiencies or limitations in code generation tools, such as software and hardware compatibility. Which could hinder development productivity and quality.
- **Insufficient Hardware and Tools for Maintenance:** Inadequate hardware or tools for maintenance pose risks to project sustainability and long-term supportability.

By identifying and understanding these potential risks, we can proactively implement strategies to mitigate their impact and ensure the successful execution of our software development project.

3.2 Risk analysis

Risk Analysis		
Risk	Probability	Effects
Power failure	3	3
data security	1	5
Bad environment when testing	4	5
Delayed development	3	4
Miscommunication	5	3
Ill staff (sick leave)	5	1
Financial problems	1	4
Code generation tools is inefficient.	1	4
Insufficient tools for maintenance	1	4
Insufficient hardware	1	4
Customer complaint	3	3
Incorrect calculation of time	3	4

3.3 Risk planning

Risk Planning	
Risk	Strategy
Power failure	Implement an uninterruptible power supply (UPS) system along with regular testing to ensure seamless transition during power failures. Additionaly, a solar power system can be a reliable soruce as a backup during power failures. This approach not only provides an environmentally friendly backup but also reduces dependence on traditional power sources.
Data security aka hacked	Enhance cybersecurity measures with regular security audits, penetration testing, and employee training. Regularly update and patch all software to address potential weak points
Bad environment when testing	Make sure to have good lighting and sufficient space for testing
Miscommunication	Improve how the team communicates by using tools like instant messaging. Make clear rules for communication and have training sessions to help everyone communicate better.
I'll staff	Train team members in different skills so everyone knows a bit about each other's jobs in case of sickness. Encourage everyone to share what they know with the team.
Financial problems	Keep a close eye on our economic situation by having regular meetings. Have some extra money set aside just in case and regularly check our budget
Code generation tools are inefficient.	Check if our code tools are working well regularly. Look for better tools if ours are not efficient. Stay connected with other developers to know about new tools. Try new tools before using them all the time to make sure they work well.
Insufficient tools for maintenance	Analyses of the tools that are in use and getting new tools if needed
Insufficient hardware	Keep a list of all our hardware and check if it's working well. Know where to get new hardware quickly if we need it. Have a plan for when we urgently need new hardware.
Customer complaints	Make a plan to talk to the customer regularly. Tell them how the project is going and listen to what they have to say. Use different ways to talk to them. Fix any problems in communication quickly to keep a good relationship with the customer.
Incorrect calculation of time	Check our estimation carefully. Have others look at our calculations to make sure they're right. Use computer tools to check our estimation when we can. Keep good records of all our documentations from the start to the end of the project.

4 Hardware and Software Requirements

4.1 Hardware Requirements

- **REQ:** Required hardware rotation motor, descent, and ascending motor, color sensor, and distance sensor.
- **REQ:** Regular maintenance for the robot, look for damaged parts and service.

4.2 Software Requirements

- **REQ01:** The robot shall locate, handle, and pick up items from a designated position without dropping them accidentally or harming the packages.
- **REQ02:** The robot should place packages at a designated position safely, gently, and reliably with reasonable speed.
- **REQ03:** The robot should be able to differentiate between workers and packages. As well, it should be able to find the packages in a designated pickup zone.
- **REQ04:** The robot should calibrate for a maximum of three different colours (except safety colour) and tell what colour the package is at a designated position.
- **REQ05:** The robot should safely, gently, and reliably place down the different items at designated drop-off zones based on the colour of the item.
- **REQ06:** The robot should be able to pick up items from varying elevated positions within the limitations of the robot at designated pickup positions.
- **REQ07:** Robot should be able to warn and communicate by any viable method to prevent collision with forklifts and other robots.
- **REQ08:** Should periodically check the pickup location to see if new items have arrived at designated pickup positions.
- **REQ09:** The Robot should be able to sort items at specific times as required by the customer.
- **REQ10:** The customer should be able to manually set the locations and height of one pickup-zone and two drop-off zones.
- **REQ11:** There should be a reliable method to stop the robot in case of an emergency.

4.3 Unable to meet certain requirments

After researching and inspecting the hardware for the provided robot. It is conlcluded that the follwing requirments won't be satisfied (in the intended way).

- **REQ03:** The robot should be able to differentiate between workers and packages. As well, it should be able to find the packages in a designated pickup zone
- **REQ04:** The robot should calibrate for a maximum of three different colours (except safety colour) and tell what colour the package is at a designated position.

The current hardware has to pick up items before it can identify them. Therefore for safety the robot needs to be operating in a specific location, where no workers should be able to enter. Example small fence around the robot, with openings for the package zones. Also Req04 won't need a safety color.

5 Robot design

5.1 Idea visualisation

We developed preliminary prototype sketches for the robot's operation. Our concept included incorporating distance and color sensors to detect objects in its path, particularly at ground level. This setup would enable the robot to distinguish between workers and packages based on the color of their work clothes.

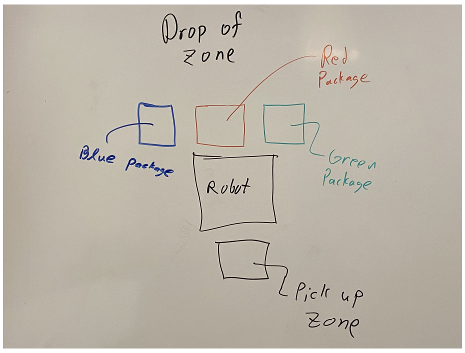


Figure 1: Design illustration sketch

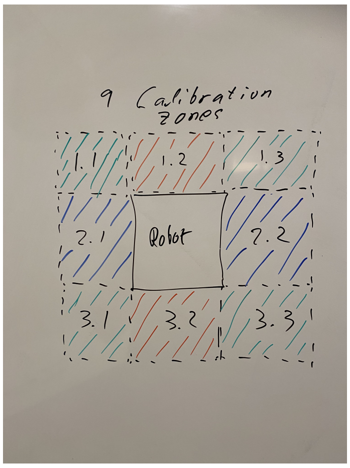
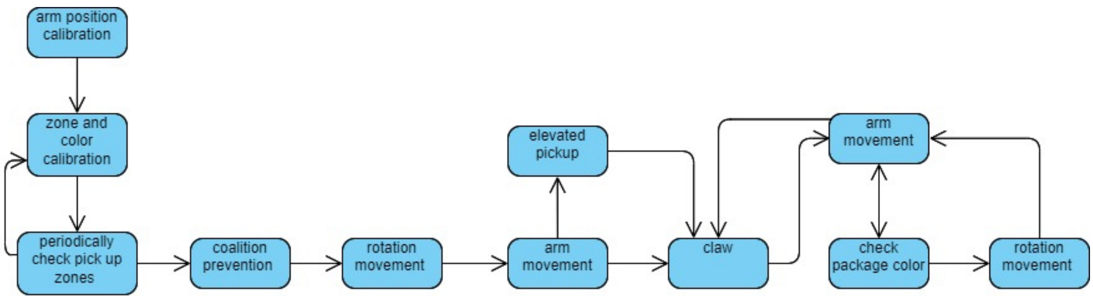


Figure 2: Zone configuration sketch

After examining the hardware and operating environment, we concluded that it would be more practical to determine zone locations based on specific angles rather than having predetermined zones. This approach eliminates the hassle of intended zones ending up in the middle between two predetermined zones, thus simplifying the calibration process for the sorting configuration.

5.2 Component diagram



6 Work Breakdown

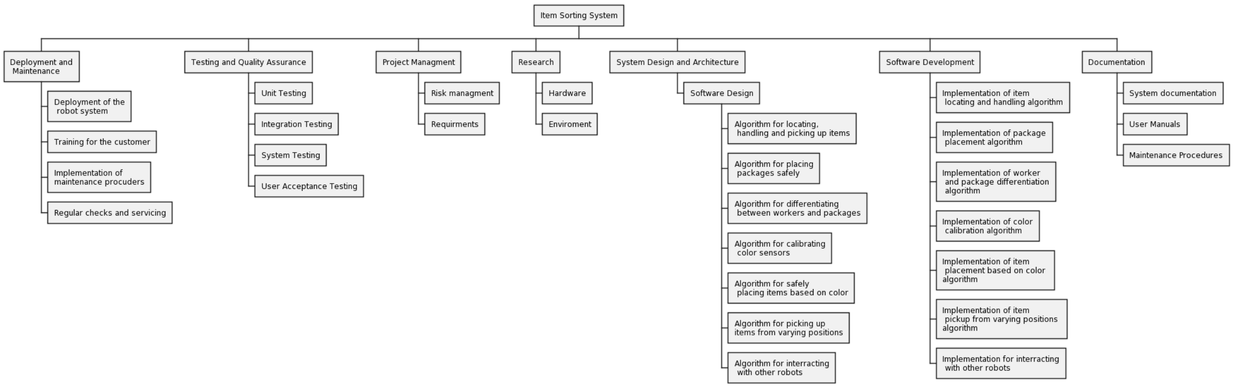


Figure 3: WBS showing the hierarchical breakdown of project tasks

7 WBS Explained

Hardware

Investigate and gather information on the required hardware components for the robot.

Product Environment

Analyze the interaction between the robot and the environment and research the conditions under which the robot works best.

Locating, Handling, and Picking up Items

Define algorithms for the robot to locate items. Develop methods to safely handle and pick up items.

Placing Packages Safely

Design a system to ensure the safe and precise placement of items.

Differentiating Between Workers and Packages

Ensure the safety of both humans and packages by making the sensors distinguish between human workers and packages.

Calibrating

Ensure the accuracy of the robotic system's sensors by implementing calibration procedures.

Safely Placing Items Based on Color

Categorize items based on color by creating algorithms and updating the sensors.

Picking up Elevated Packages

Design mechanisms to allow the robot to pick up packages from elevated surfaces.

Rotation Algorithm

Design a rotation plan to allow the robot to rotate during the sorting process.

Requirement Refining

Refine and clarify project requirements for better understanding.

Risk Management

Identify and set risk management strategies to manage potential risks associated with the project.

Item Locating and Handling Algorithm

Develop algorithms that allow the sensors within the robot to accurately locate items within the environment.

Package Placement Algorithm

Ensure the placement is secure, organized, and aligns with the overall sorting strategy based on colors.

Worker and Package Differentiation Algorithm

Enhance safety by working on algorithms to differentiate between human workers and packages.

Color Calibration Algorithm

Implement algorithms to calibrate the robot's sensor for accurate color recognition.

Item Placement Based on Color Algorithm

Develop algorithms that guide the robot in placing items in designated locations based on their colors using color-coded criteria.

Item Pickup from Varying Positions Algorithm

Code algorithms to enable the robot to pick up items from various positions, ensuring the package is in the right zone to be picked up.

Unit Testing

Test each unit of the software to verify that it performs its specific function correctly.

Integration Testing

Examine how various software components work together, ensuring different functions seamlessly interact without causing errors.

System Testing

Assess the complete robotic system to verify that it performs all the intended tasks, addressing the interplay between hardware and software.

Acceptance Testing

Validate that the product meets the customer's specified requirements and is ready for deployment.

Deployment of the Robot System

Implement the robot system in the warehouse.

Training for the Customer

Provide training for the customer to use and maintain the robot.

Implementation of Maintenance Procedures

Develop procedures for regular maintenance checks.

Regular Maintenance Checks and Servicing

Regularly inspect the robot for optimal performance.

System Documentation

Document the system's architecture, components, and functionalities.

User Manuals

Create user manuals for the operators and end-users.

Maintenance Procedures

Document procedures for maintaining the robot.

8 Gantt Chart

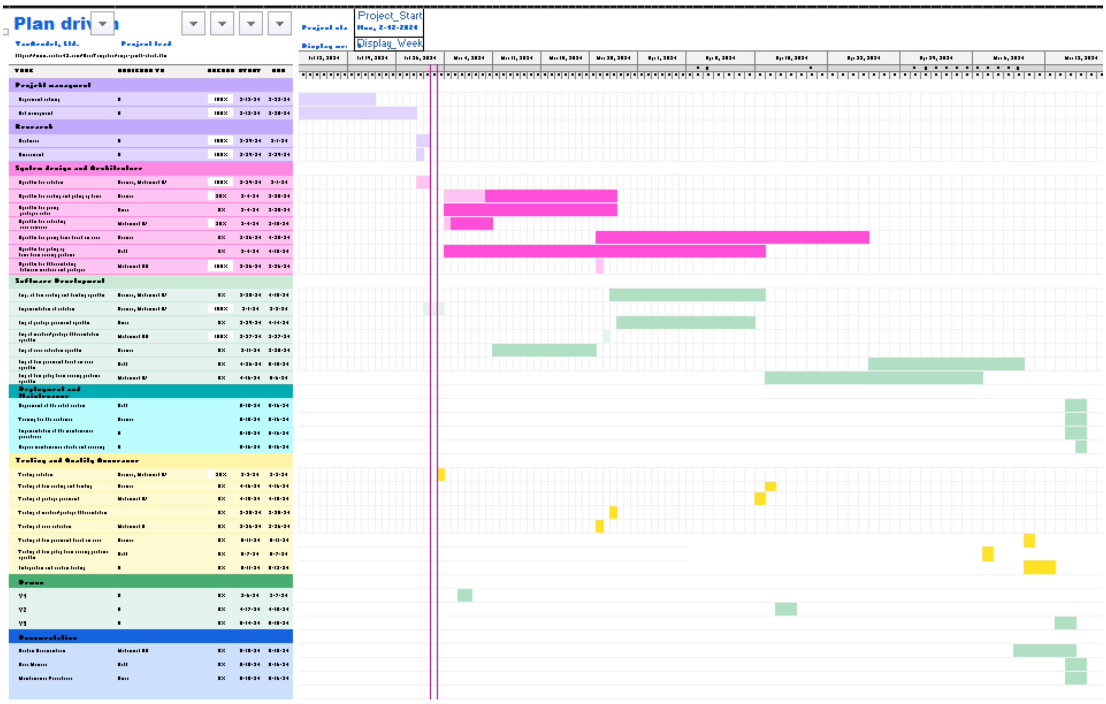


Figure 4: Gantt Chart: a snapshot of the project's schedule and milestones.

Milestones

Demo One

Successfully creating a functional rotating arm.This milestone includes the execution of algorithms essential for the arm's rotation successfully, marking a critical step in our project's development.

Demo Two

Developing the capability to locate, pick up, handle packages, and safely place them, along with the successful calibration of the color sensor of package colors.

Demo Three

Having the arm to place packages in their designated colored zones, coupled with the added functionality of executing this task seamlessly from elevated positions. The communication between the robot arms should also be complete in this demo

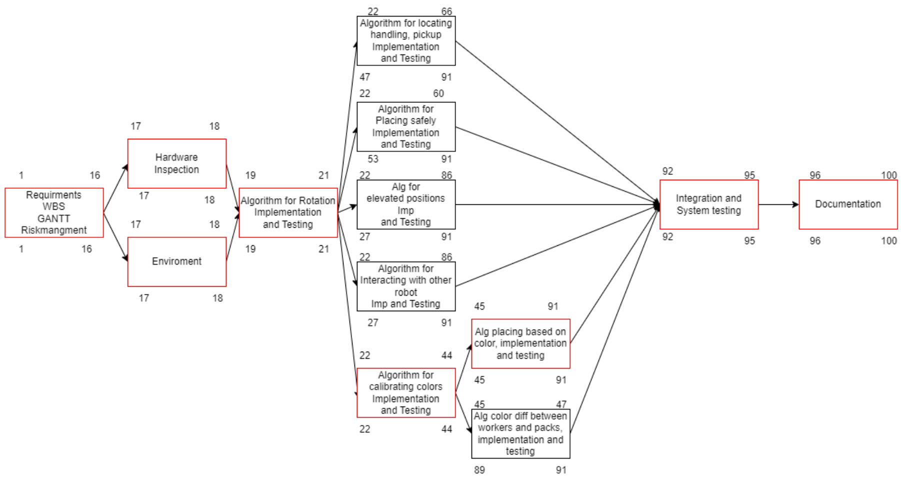
9 Testing Strategy

Our testing strategy has unit testing initially after each implementation phase to verify that individual units meet their specifications to ensure the early detection and resolution of any potential issue.

Our group actively engages in defect testing for defects in our code to debug. This approach allows us to enhance the overall code quality.

At the end, we perform thorough system testing to validate the integration of all components, to ensure their functionality, mitigating any potential conflicts between different parts and conforming that project requirements have been met.

10 Critical Path



11 Estimation and resource allocation

Our estimation process was methodical, ensuring accuracy and thoroughness in forecasting project timelines. We began by constructing a Work Breakdown Structure (WBS), breaking the project into manageable tasks. This hierarchical breakdown facilitated a granular examination of each aspect, aiding in the estimation of their durations.

WBS Breakdown

The WBS delineated the project into smaller, more easily estimable tasks. This breakdown allowed us to delve into the specifics of each task, considering its complexity and requirements. By dissecting the project into manageable units, we gained clarity on the scope and intricacies, facilitating precise time estimation.

Collaborative Task Allocation

To optimize our estimation process, we leveraged the expertise and strengths of each team member. Tasks were distributed among team members based on their individual proficiencies and insights. This collaborative approach ensured a comprehensive assessment of task durations, considering diverse perspectives and skill sets.

Time Allocation

Our estimation assumed an allocation of 2 hours per day per team member for the project. This standardized approach provided a consistent basis for estimating task durations across the project timeline.

Implementation in Gantt Chart:

The estimation outcomes were seamlessly integrated into the Gantt chart, serving as a foundational element in project planning. Each task's estimated duration was meticulously documented, providing a visual representation of the project timeline. An example of this integration is illustrated in the accompanying image, showcasing the collaborative efforts and time allocation for specific tasks.

Methodology Conclusion

Through a systematic approach encompassing WBS breakdown, collaborative task allocation, and standardized time allocation, we achieved a robust estimation of the project timeline. This methodology ensured accuracy and realism in forecasting, laying a solid foundation for successful project execution.

12 Resource Allocation

In the resource allocation, our plan encompassed a substantial amount of group work on campus. While certain tasks were undertaken individually, others necessitated collaboration among multiple individuals. Initially, we engaged in discussions to identify each person's strengths, subsequently distributing tasks accordingly. Upon completion of a task, the responsible member presented their work, and team members provided constructive feedback. For instance, within the documentation section, a specific task pertaining to system documentation was assigned to an individual with a stipulated timeframe of two days, constituting a total workload of 4 hours.

Deocumentation				
System Documentaion	Mohamad Al Darwich	0'%	4/5/24	4/6/24

Conversely, the Project Management section required a more extensive effort, prompting the collaboration of all the members on the same task, namely 'Requirement Refining. All remaining details are documented in the Gantt chart, which is submitted along with the LaTeX file.

Projekt managment				
Requirment refining	Everybody	100%	2/12/24	2/18/24
Risk managment	Everybody	100%	2/12/24	2/18/24