

# GOOD MORNING!

早上好!

안녕하세요!

---

PROJECT INTRODUCTION



# DAY I (DONE)

---

- Welcome
- Project Introduction
- Introduction to Project Development Process
- Business Requirement Development
- System Requirement Development
- System(High Level) Design
- Time Management

# DAY 2 (DONE?)

---

- YOLOv8 기반 데이터 수집/학습/deploy (Detection Alert)
  - 감시용 데이터 수집(bus, truck, tank 등)
  - 감시용 데이터 라벨링
  - YOLOv8 기반 학습
  - YOLOv8 Object Detection
- Porting to ROS
  - Create Detection Alert Node
  - Generate Topics to send image and Obj. Det. results
  - Create Subscriber node and display image and print data from the Topic

# DAY 3

---

- AMR (Autonomous Mobile Robot)기반 카메라 인식 autonomous driving 시스템 with obstacle avoidance 구축 (AMR Controller)
  - Digital Mapping of environment
  - Goal Setting and Obstacle Avoidance using Navigation
  - Object Tracking w/ AMR camera
  - Control logic between navigation/obj. tracking/ obj. following (teleop)
- Porting to ROS
  - Create AMR Controller Node
  - Create and send Obj.Tracking Image and data to Sysmon
- And finally, Integration and Test of Detection Alert & AMR Controller

# DAY 4

---

- Flask 를 이용한 웹 서버 구축 (System Monitor)
  - Flask/HTML Intro
  - Deploy YOLOv8 Obj. Det results to web
  - Log in 기능 구현
  - Sysmon 웹기능 구현
  - 알람 기능 구현
- SQLite3를 이용한 데이터베이스 구축 및 연동 (System Monitor)
  - SQLite3 기본 기능 구현
  - DB 기능 구축
  - 알람이 울리는 경우 DB에 저장하는 기능 구현
  - 저장된 내용 검색하는 기능 구현



# DAY 4

---

- Porting to ROS
  - Update Sysmon Node code
  - Update the database with received Obj. Det. Data from Detection Alert Node
  - Display the content of DB on System Monitor web page
- And finally, Integration and Test of Detection Alert & System Monitor

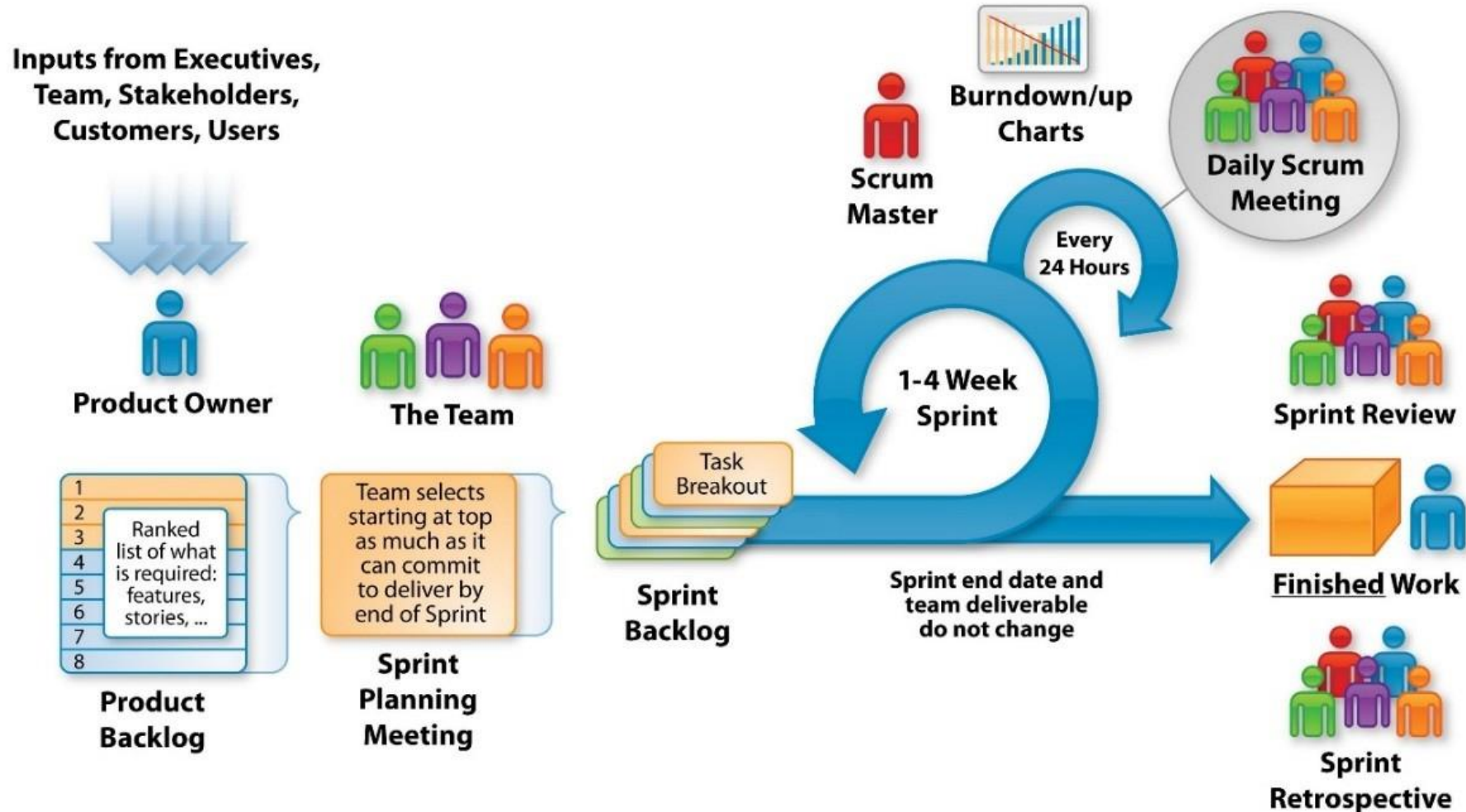
프로젝트 RULE NUMBER ONE!!!

---

Have Fun Fun Fun!



# The Agile - Scrum Framework

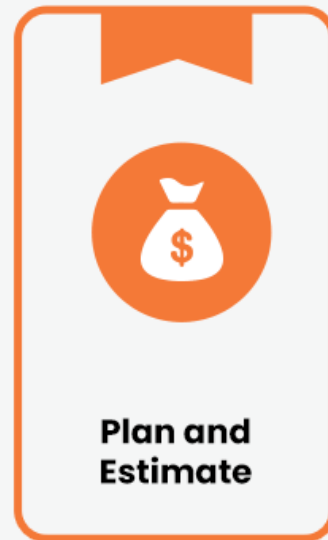




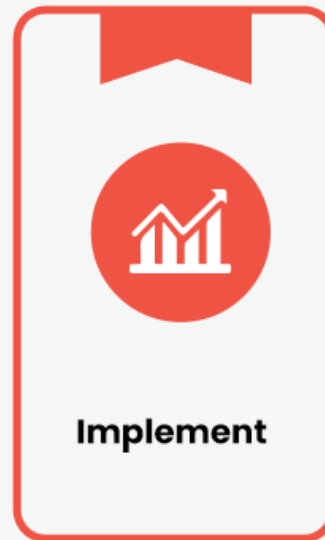
# 5 Stages of Scrum Sprint



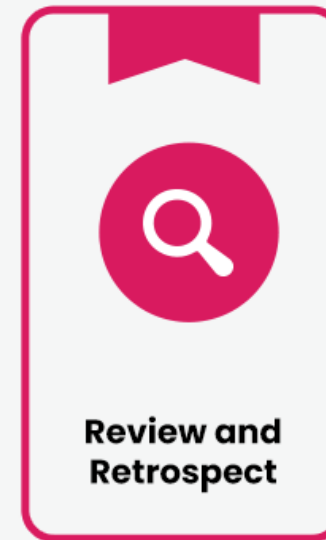
This phase includes the processes related to the commencement of a project, such as a scope and objectives, creating and distributing its charter, and taking other steps to guarantee success.



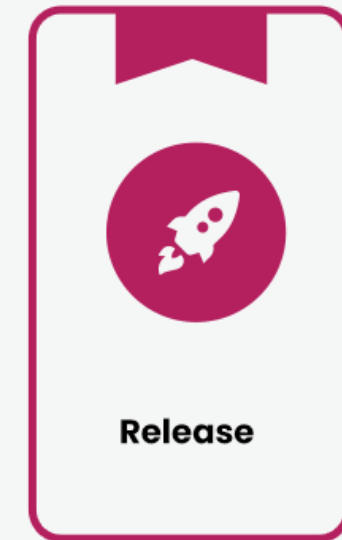
This phase involves planning and estimating processes, including creating user stories, approving, assessing, committing user stories, creating tasks, evaluating tasks, and creating a Sprint backlog.



This phase is about executing the tasks and activities to create a product. These activities include building the various outputs, conducting daily standup meetings, and grooming the product backlog.

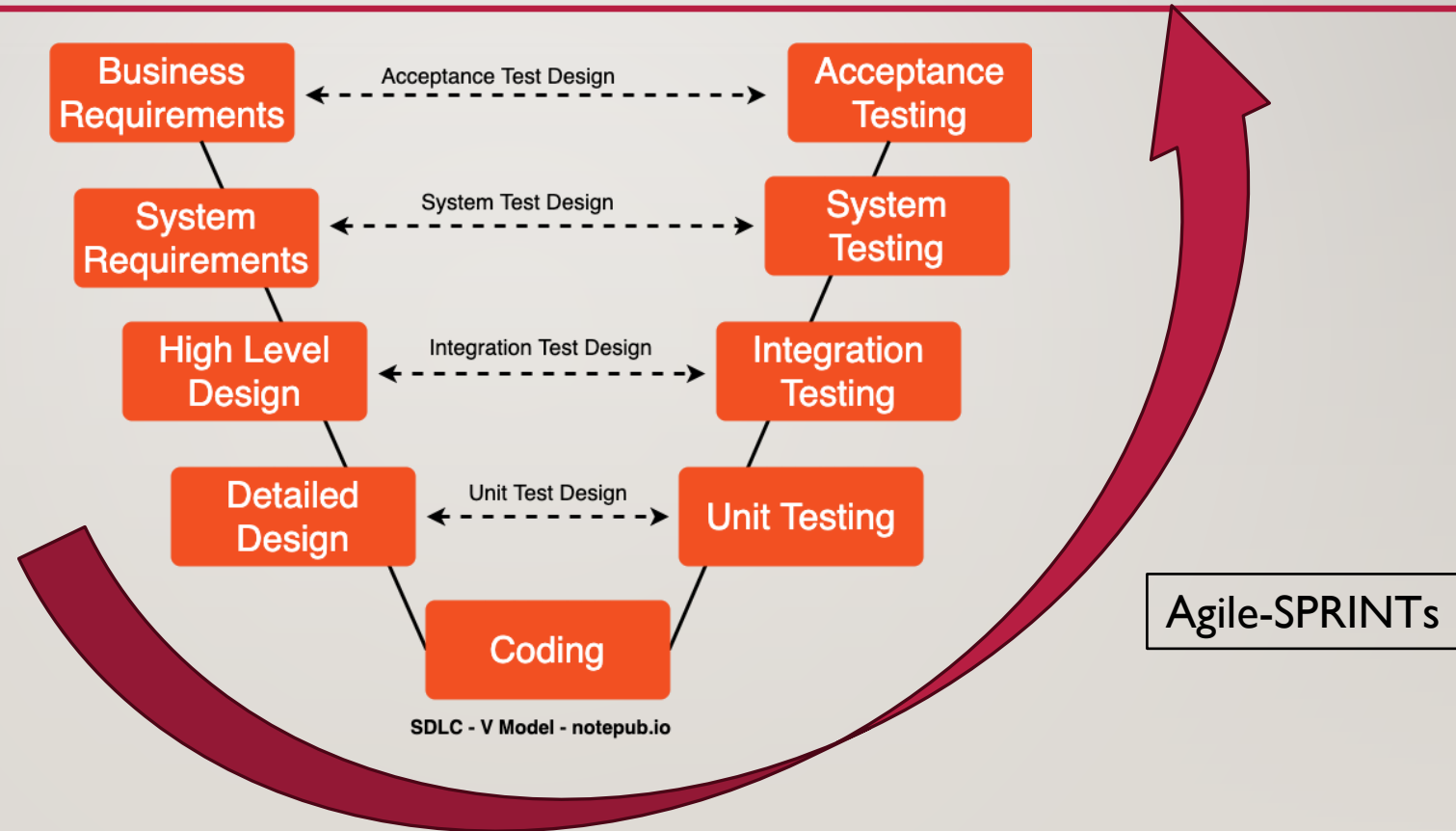


This stage of the project lifecycle is concerned with evaluating what has been accomplished so far, whether the team has worked to plan, and how it can do things better in the future.



This stage highlights delivering the accepted deliverables to the customer and determining, documenting, and absorbing the lessons learned during the project.

# SW DEVELOPMENT PROCESS



# PROJECT SPRINTS

- Detection Alert

- Camera Capture
- Object Detection
- Send messages to other subsystems

- AMR Controller

- Receive messages and act accordingly
- Move using (SLAM) with Obstruction avoidance
- Target Acquisition (Obj. Det.) and Tracking
- Follow target using camera and motor control

- System Monitor

- Receive and Display Detection Camera and info
- Receive and Display AMR Camera and info
- Store, display, and report Information and Alerts

# TEAM EXERCISE 5

---

Perform coding and testing of Detection Alert Module



# RESULTS & CODE REVIEW BY EACH TEAM

---

Show actual results against the expected results and explain the code written

# PROJECT SPRINTS

- Detection Alert

- Camera Capture
- Object Detection
- Send messages to other subsystems

- AMR Controller

- Receive messages and act accordingly
- Move using (SLAM) with Obstruction avoidance
- Target Acquisition (Obj. Det.) and Tracking
- Follow target using camera and motor control

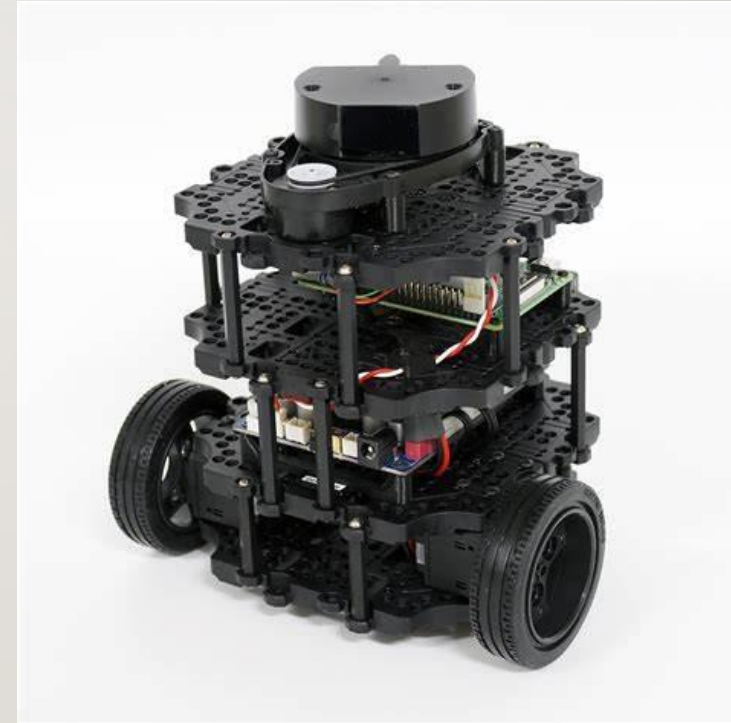
- System Monitor

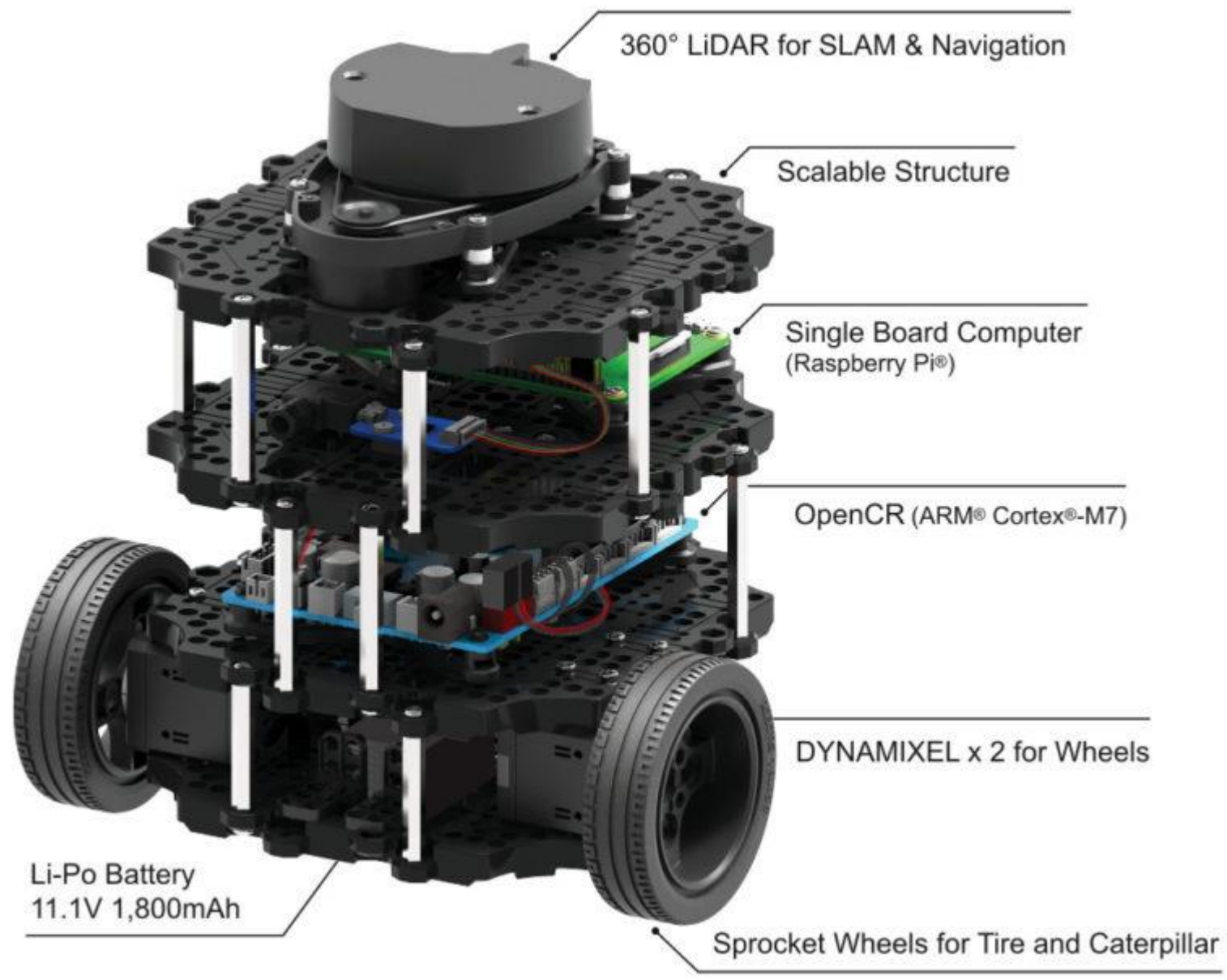
- Receive and Display Detection Camera and info
- Receive and Display AMR Camera and info
- Store, display, and report Information and Alerts

# INTRODUCTION TO AMR

---

- [TurtleBot3](https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/)
- <https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/>



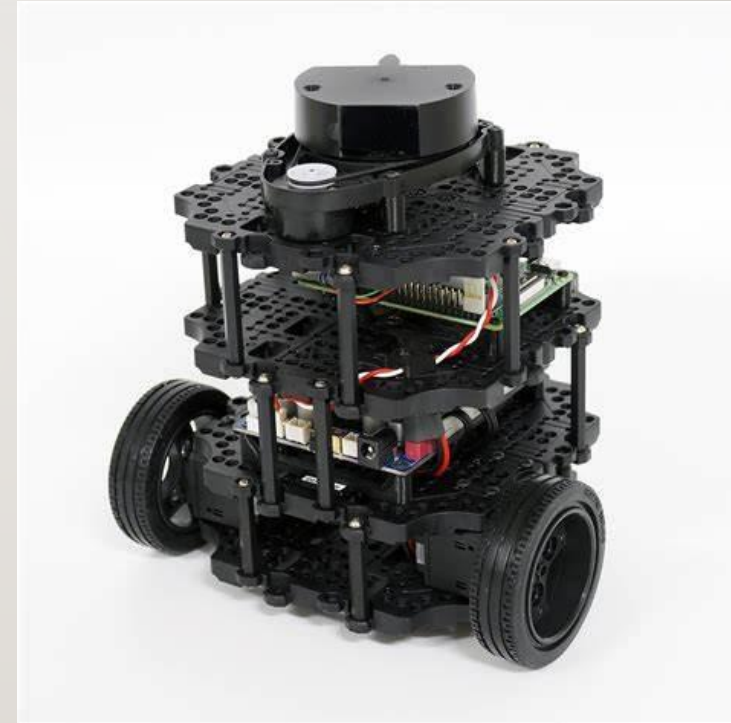




# AMR DEMO

---

- [TurtleBot3](https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/)
- <https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/>
- Navigation with SLAM
- Teleop with keyboard



# POWER ON AND OFF AMR (VERY IMPORTANT!!!!)

---

## POWER ON

- Make sure both Jetson and OpenCR is powered off.
- Must turn on OpenCR first!!!!
- Wait for the music from OpenCR
- Finally turn on Jetson

## POWER OFF

- Execute proper shutdown

\$ sudo shutdown now

Turn off Jetson, Turn off OpenCR

Do **NOT INSTALL** any packages to AMR **WITHOUT** speaking with me first!!!

# SETUP PC FOR AMR

---

## INSTALL DEPENDENT ROS 2 PACKAGES

\$ Check if installed

\$ apt list | grep gazebo

\$ apt list | grep carto

If not,

\$ sudo apt install ros-humble-gazebo-\*

\$ sudo apt install ros-humble-cartographer

\$ sudo apt install ros-humble-cartographer-ros

## TURTLEBOT3

<https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/>

# SETUP PC FOR AMR

---

## CHECK IF ALREADY INSTALLED

```
$ apt list | grep dynam
```

```
$ apt list | grep turtlebot3
```

## INSTALL TURTLEBOT3 PACKAGES

If not already installed,

```
$ source ~/.bashrc
```

```
$ sudo apt install ros-humble-dynamixel-sdk
```

```
$ sudo apt install ros-humble-turtlebot3-  
msgs
```

```
$ sudo apt install ros-humble-turtlebot3
```



# SETUP PC FOR AMR

---

- If you want to download the source code

```
$ mkdir -p ~/turtlebot3_ws/src
```

```
$ cd ~/turtlebot3_ws/src/
```

```
$ git clone -b humble-devel  
https://github.com/ROBOTIS-  
GIT/DynamixelSDK.git
```

```
$ git clone -b humble-devel  
https://github.com/ROBOTIS-  
GIT/turtlebot3_msgs.git
```

```
$ git clone -b humble-devel  
https://github.com/ROBOTIS-GIT/turtlebot3.git
```

```
$ cd ~/turtlebot3_ws
```

```
$ colcon build --symlink-install
```

```
$ echo 'source  
~/turtlebot3_ws/install/setup.bash' >> ~/.bashrc
```

```
$ source ~/.bashrc
```



# SETUP ROS ID

---

## PC

```
$ cat ~/.bashrc
```

```
$ source ~/.bashrc
```

```
$ env | grep ROS
```

If ROS\_DOMAIN\_ID does not exist or is not correctly set,

```
$ echo 'export ROS_DOMAIN_ID=I  
#TURTLEBOT3' >> ~/.bashrc
```

(ID = 1,2,3,4,5 depends on your team number)

# HOW TO CONNECT TO AMR

---

## PC

\$ sudo ufw status

\$ sudo ufw disable

- disables firewall for ubuntu systems

# HOW TO CONNECT TO AMR

---

## YOUR AMR IS A HOTSPOT

- Connect to wifi:
  - Turtlebot3\_AP\_<n>
- Connect via ssh

## PC TERMINAL

```
$ dpkg -l | grep openssh
```

If not installed...

```
$ sudo apt install openssh-server -y
```

```
$ ssh -X rokey<n>@<ip_address>
```

\*allows Vscode/Rviz to run



# HOW TO CONNECT TO AMR

---

## PC TERMINAL

```
$ ros2 run demo_nodes_cpp talker
```

## SSH AMR TERMINAL


```
$ cat ~/.bashrc
```

If ROS\_DOMAIN\_ID is not correctly set,

```
$ echo 'export ROS_DOMAIN_ID=  
#TURTLEBOT3' >> ~/.bashrc
```

(ID = 1,2,3,4,5 depends on your team number)

```
$ ros2 run demo_nodes_cpp listener
```



# CREATE YOUR WORKSPACE UNDER AMR \$HOME DIRECTORY

---

## SSH AMR TERMINAL

```
$ mkdir  
~/rokeyl_<grp_letter><grp_num>_ws  
(i.e. mkdir ~/rokeyl_A2_ws)
```

- Put all your file under this directory and remove at the end of the class
- Delete the directory at the end of the class

# SETTING UP TO USE TURTLEBOT3

---

## PC TERM I

```
$ ssh -X  
rokey<n>@<ip_address>
```



## SSH AMR TERM I

```
$
```

# SETTING UP TO USE TURTLEBOT3

---

## SSH AMR TERM I

```
$ cat ~/.bashrc
```

Check if export command on the right  
is in the .bashrc, if not execute the  
command on the right

## SSH AMR TERM I

```
$ echo 'export  
TURTLEBOT3_MODEL=burger' >>  
~/.bashrc
```



# SETTING UP TO USE TURTLEBOT3

---

## PC TERM 1

```
$ cat ~/.bashrc
```

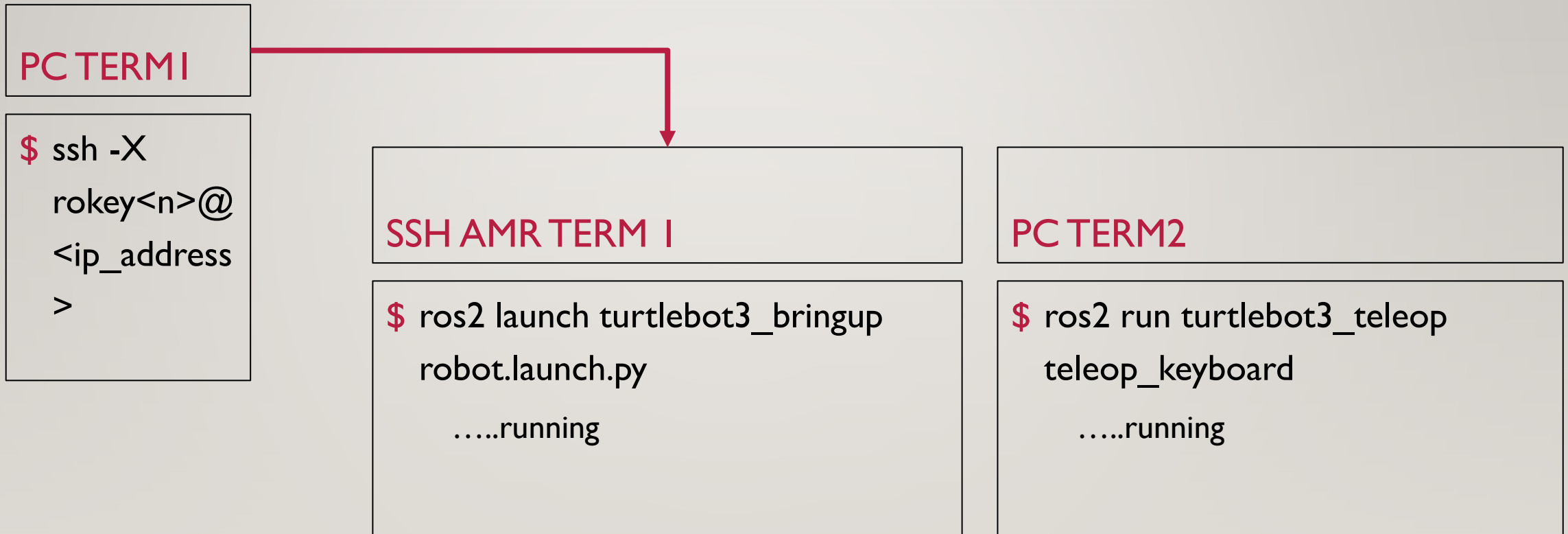
Check if export command on the right is in the .bashrc, if not execute the command on the right

## PC TERM 1

```
$ echo 'export  
TURTLEBOT3_MODEL=burger' >>  
~/.bashrc
```

# USING AMR TELEOP

---



# DIGITAL MAPPING

---

## STEP1: SSH AMR TERM 1

```
$ source ~/bashrc  
  
$ ros2 launch turtlebot3_bringup  
  robot.launch.py
```

## STEP2: PC TERM 2

```
$ source ~/bashrc  
  
$ ros2 launch turtlebot3_cartographer  
  cartographer.launch.py
```

# DIGITAL MAPPING

---

## STEP1: SSH AMR TERM 1

```
$ ros2 launch turtlebot3_bringup  
  robot.launch.py  
  
..... running
```

## STEP2: PC TERM 2

```
$ ros2 launch turtlebot3_cartographer  
  cartographer.launch.py  
  
..... running
```

## STEP3: PC TERM 3

```
$ source ~/bashrc  
  
$ ros2 run turtlebot3_teleop  
  teleop_keyboard
```



# DIGITAL MAPPING

---

## STEP1: SSH AMR TERM 1

```
$ ros2 launch turtlebot3_bringup  
  robot.launch.py  
  
..... running
```

## STEP2: PC TERM 2

```
$ ros2 launch turtlebot3_cartographer  
  cartographer.launch.py  
  
..... running
```

## STEP3: PC TERM 3

```
$ ros2 run turtlebot3_teleop  
  teleop_keyboard  
  
..... running
```

## STEP4: PC TERM 4(AT THE END)

```
$ source ~/bashrc  
  
$ ros2 run nav2_map_server  
  map_saver_cli -f ~/<my_dir>/map
```

# DIGITAL MAPPING

---

## CHECK IF CORRECT

\$ xdg-open <map-path>/map.pgm

Or,

\$ eog <map-path>/map.pgm

# NAVIGATION W/ MAP

---

## STEP1: SSH AMR TERM 1

```
$ source ~/bashrc  
  
$ ros2 launch turtlebot3_bringup  
  robot.launch.py
```

## STEP2: PC TERM 2

```
$ source ~/.bashrc  
  
$ ros2 launch turtlebot3_navigation2  
  navigation2.launch.py  
  map:=<map_file_path> (i.e:  
    $HOME/my_dir/map/map.yaml)
```

# NAVIGATION W/ MAP

---

## STEP1: SSH AMR TERM 1

```
$ ros2 launch turtlebot3_bringup  
  robot.launch.py  
....running
```

## STEP2: PC TERM 2

```
$ ros2 launch turtlebot3_navigation2  
  navigation2.launch.py  
  map:=<map_file_path> (i.e:  
    $HOME/my_dir/map/map.yaml)  
....running
```

\*Perform 2D Pose Estimate and Send Goal



# IF MAPPING IS DONE OF PC MOVE MAP TO AMR

---

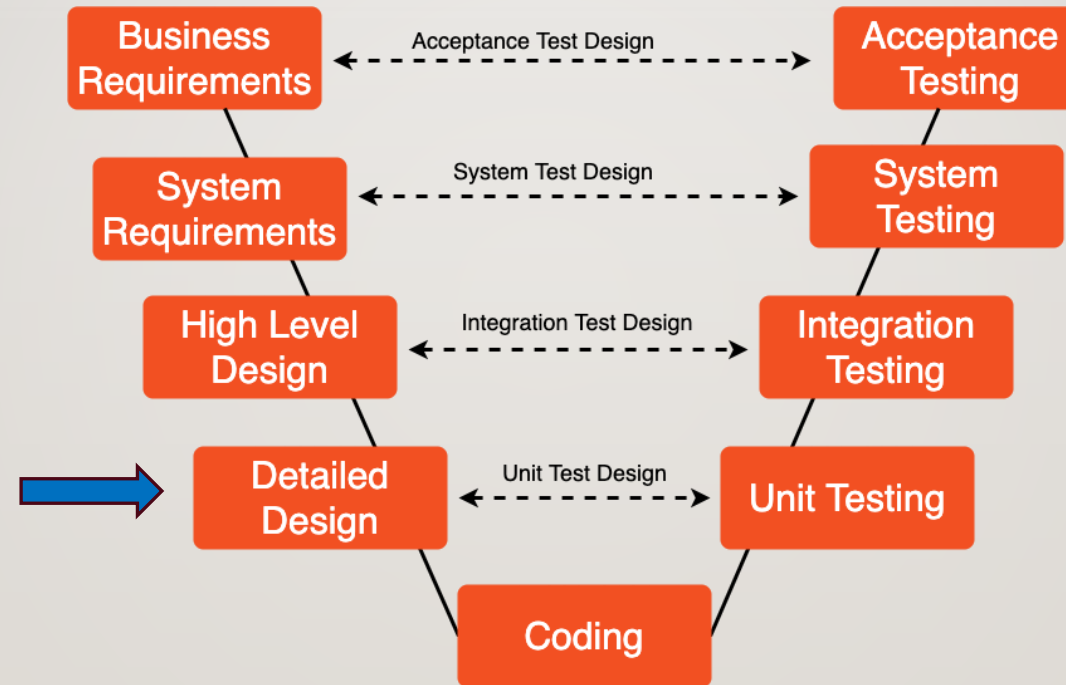
```
$ scp map.yaml map pgm rokey<n>@<rokey IP>:$HOME/map/
```

# AMR CONTROLLER SPRINT

---



# SPRINT 2 – AMR CONTROLLER



SDLC - V Model - notepub.io

# TEAM EXERCISE 6

---

Perform Detail Design of AMR Controller Module using Process Flow Diagram

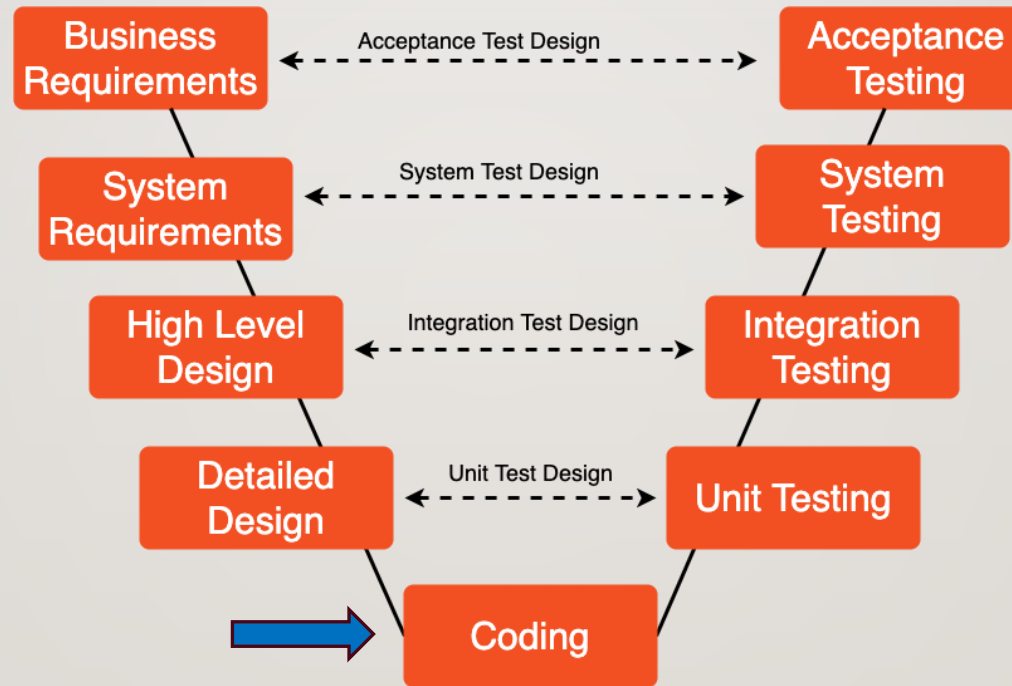


# DETAIL DESIGN REVIEW BY EACH TEAM

---

Using the process flow diagram present team's design

# SPRINT 2 – AMR CONTROLLER



SDLC - V Model - notepub.io

# SETTING UP VS CODE FOR REMOTE EDITING

---

- Install VS Code Remote - SSH Extension:
  - Open VS Code on your local machine.
  - Go to the Extensions view (Ctrl + Shift + X).
  - Search for "Remote - SSH" and install it.
- Connect to the Remote Server:
  - Press F1 or Ctrl + Shift + P to open the Command Palette.
  - Type Remote-SSH: Connect to Host and select it.
  - Enter the SSH connection string (e.g., user@hostname) and connect.
- Open a Remote Folder:
  - Once connected, VS Code will display a new window with a remote indicator in the bottom-left corner.
  - You can open any folder or file from the remote server and edit it in your local VS Code instance.

# CODING HINT

---

- Initial Pose
  - nav2
  - rviz2 – 2D estimate pose
  - ros2 topic echo /initialpose
- Sending Goals
  - nav2
  - Rviz2 – send goals
    - ActionClient
    - NavigateToPose
  - ros2 topic echo /amcl\_pose
- Stopping Navigation
  - NavigateToPose.cancel\_all\_goals\_async()
- Sending multiple goals
  - ActionClient
  - /follow\_waypoints
- Manual control of AMR Odometry
  - How to move forward, backward, left and right???

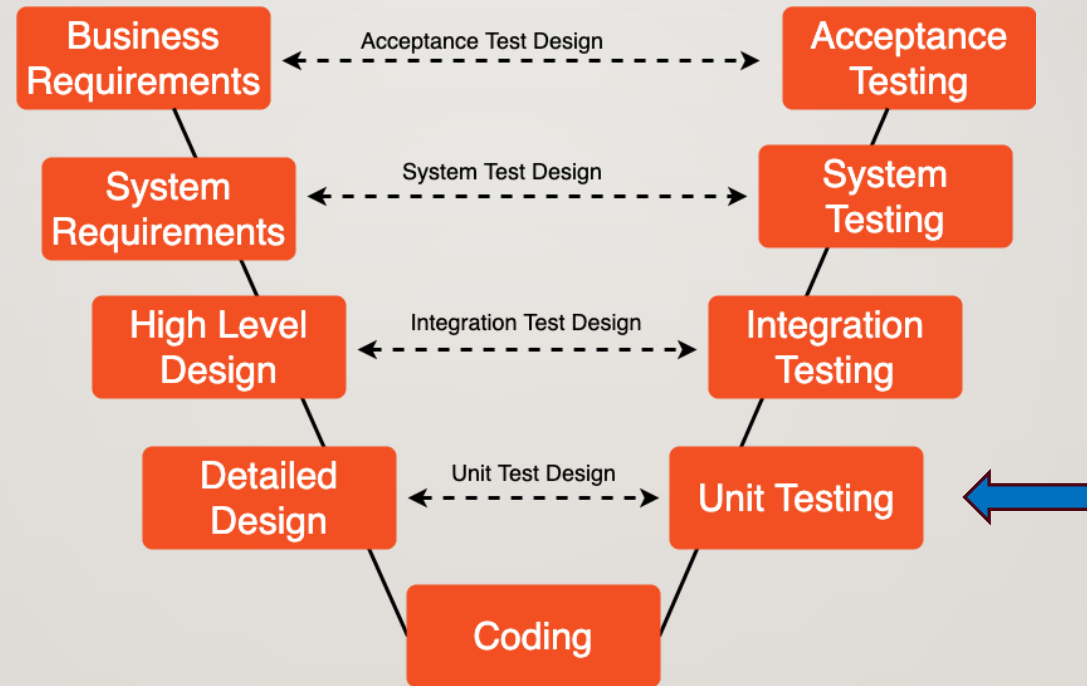


# EXPECTED OUTCOME

---

AMR navigates to avoid obstacles, ignores dummies, track, and follow target

# SPRINT 2 – AMR CONTROLLER



SDLC - V Model - notepub.io

# TEAM EXERCISE 7

---

Perform coding and testing of AMR Controller Module

# RESULTS & CODE REVIEW BY EACH TEAM

---

Show actual results against the expected results and explain the code generated





# EXPECTED OUTCOME

---

- Detection Alert and AMR Controller able to pass topics for necessary actions between

# TEAM EXERCISE 8

---

Perform integrate and test of Detection Alert and AMR Controller Modules

# RESULTS & CODE REVIEW BY EACH TEAM

---

Show actual results against the expected results and explain the code written



Send System Design Doc.  
Here:

