

GOOD MORNING!

早上好!

안녕하세요!

DAY 2



DAY 1 RECAP



2 PROJECTS

- Mini Project (Individual Team)

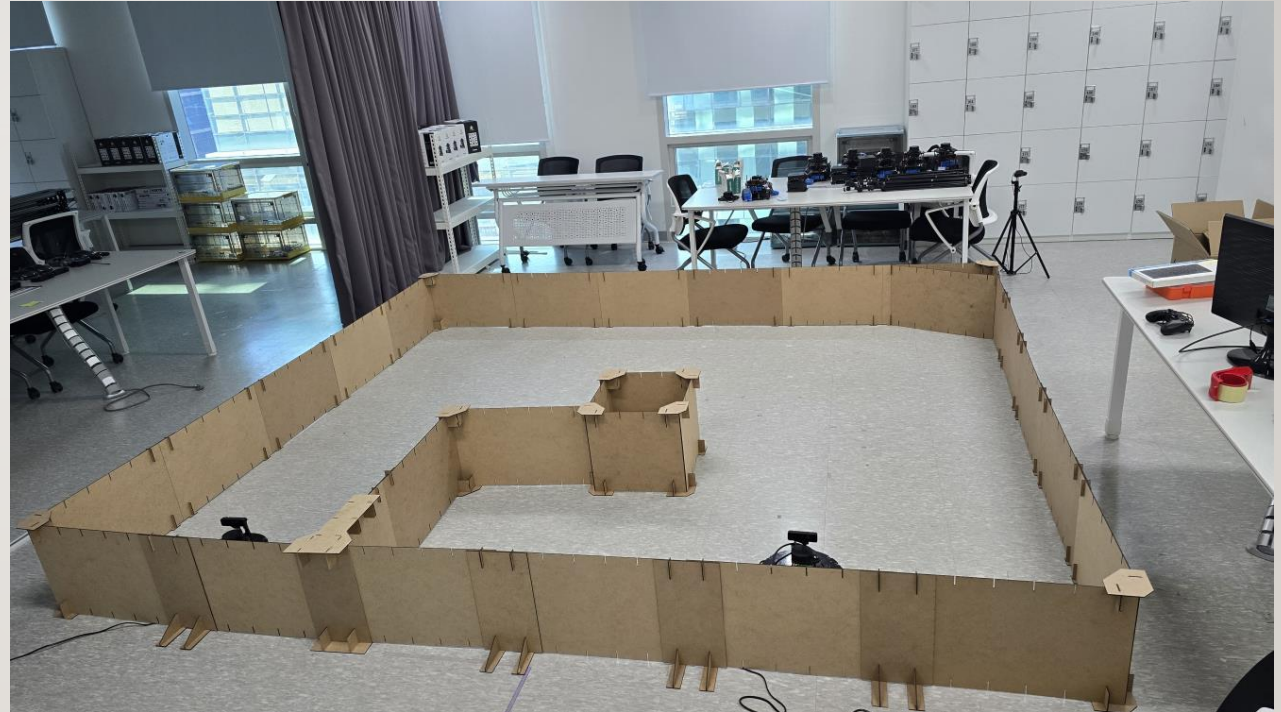
- For learning techniques

시스템 개발 프로세스의 이해 개발 환경 구축
AI VISION 기술 탐색 및 검증
로봇 AMR 제어 기술 탐색 및 검증
웹 시스템 모니터 기술 탐색 및 검증

- Final Project (2 Teams in One)

통합 시스템 설계 및 개발
시스템 발표 및 시연

MINI PROJECT DESCRIPTION



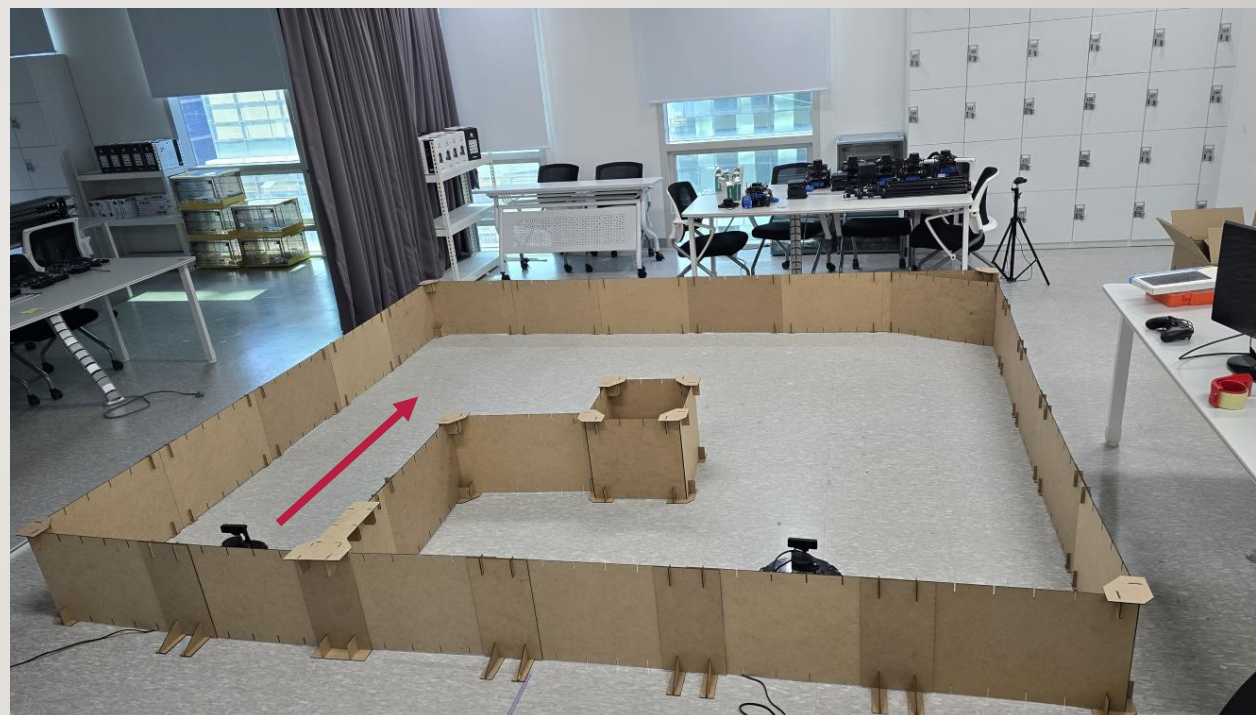
KEY SUBSYSTEM (MODULES) TO DEVELOP

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

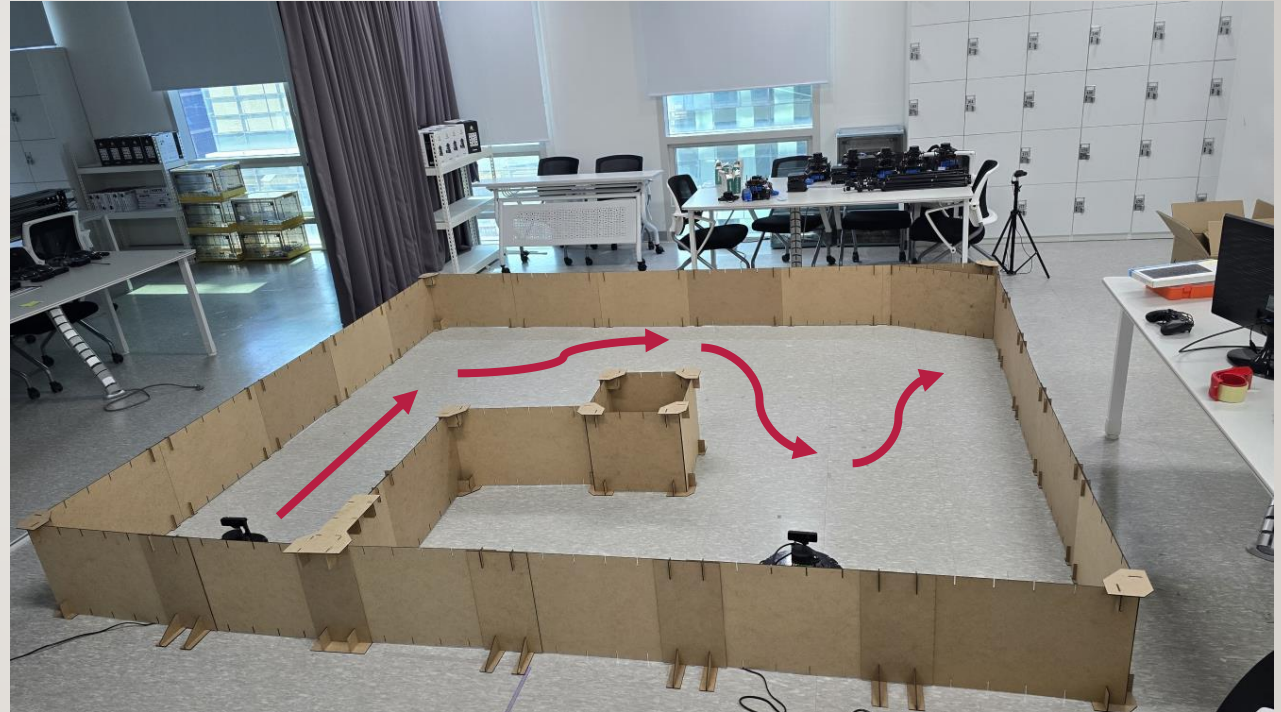
DETECTION ALERT



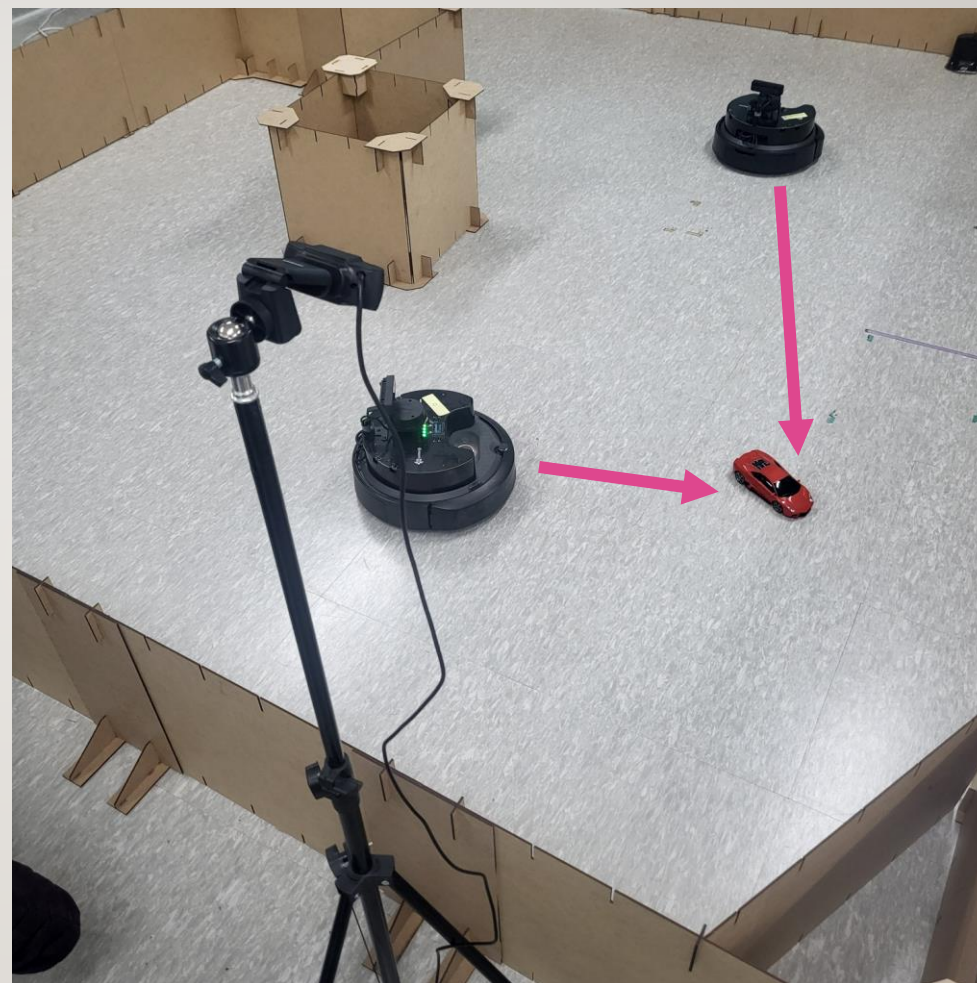
START



NAVIGATE



TRACK & FOLLOW



DAY I

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

DAY 2 (MINI PROJECT)

- Yolo객체 인식 모델 활용과 성능 평가 방법 이해
- Custom Dataset과 Fine Tuning으로 자체 객체 인식 모델 구현 및 평가
- (Optional)경량화 모델 등 개별 요구사항에 적합한 모델 탐색 및 성능 검증
- 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 (MINI PROJECT)

- AMR(Autonomous Mobile Robot)
Turtlebot4 개발 환경 구축
- 로봇 개발 환경에 완성 모델 서빙 및 테스트 / 로봇 H/W, 제반 환경의 한계점 도출
 - Tracking 데이터 수집(bus, truck, tank 등)
 - Tracking 데이터 라벨링
 - YOLOv8 기반 학습
 - YOLOv8 Object **Tracking**
- Turtlebot4 시뮬레이션 환경 구축
 - SLAM과 Map 생성 및 파라미터 튜닝 (Localization, AMCL)
 - AutoSLAM으로 맵 생성

프로젝트 RULE

80/20 → 20/80

TEAMWORK AND PROJECT MANAGEMENT



HOW TO WORK TOGETHER

- Participate, Participate, Participate!!!
- No long emails or Kakaotalk, prefer face to face
- Be open to suggestions and idea
- Be proactive, take initiative
- HOW is as important as WHAT

BRAINSTORMING RULES

- Every input is good input
- Do not critique inputs only seek to understand
- Organize inputs into logical groupings
- Sequence or show relationships as needed
- Use Posted Notes on Flip Chart



프로젝트 RULE NUMBER ONE!!!

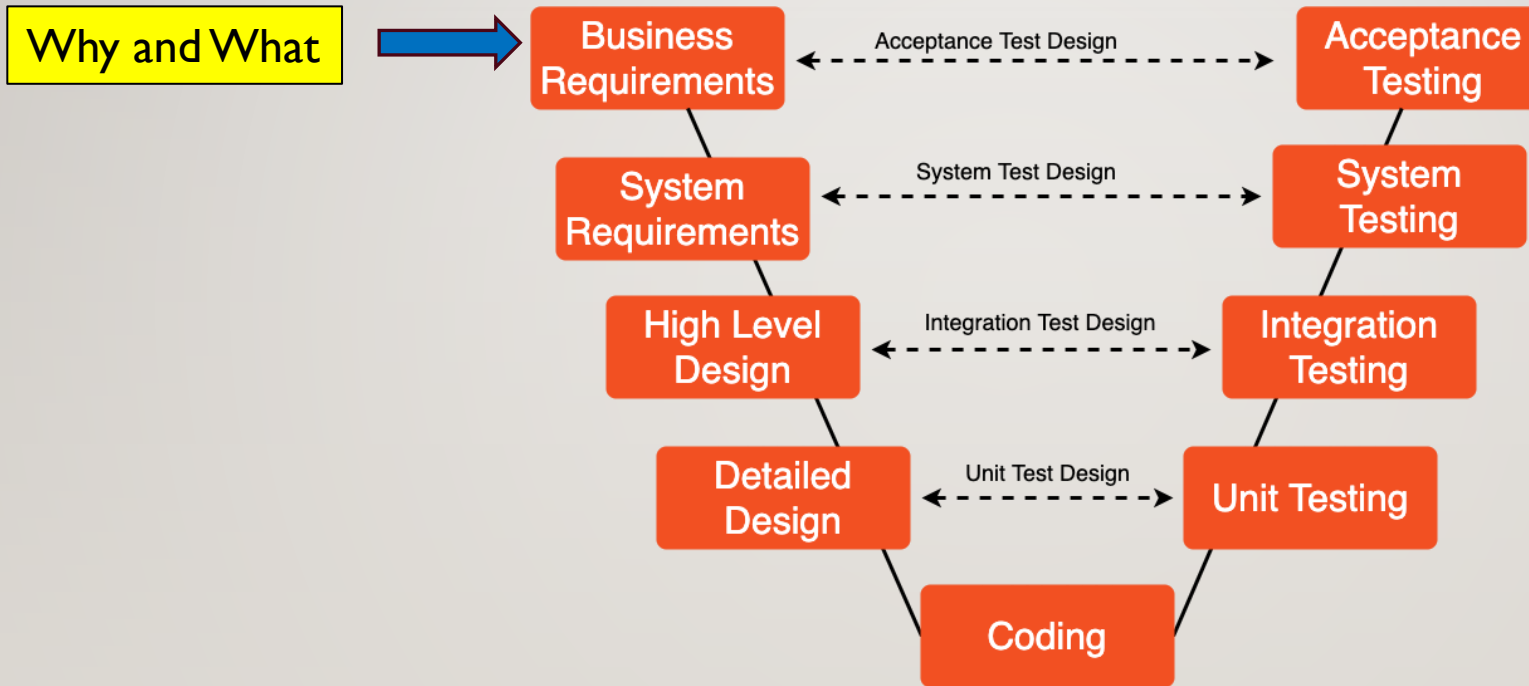
Have Fun Fun Fun!



PROJECT DEVELOPMENT
IS A PROCESS



SW DEVELOPMENT PROCESS



SDLC - V Model - notepub.io

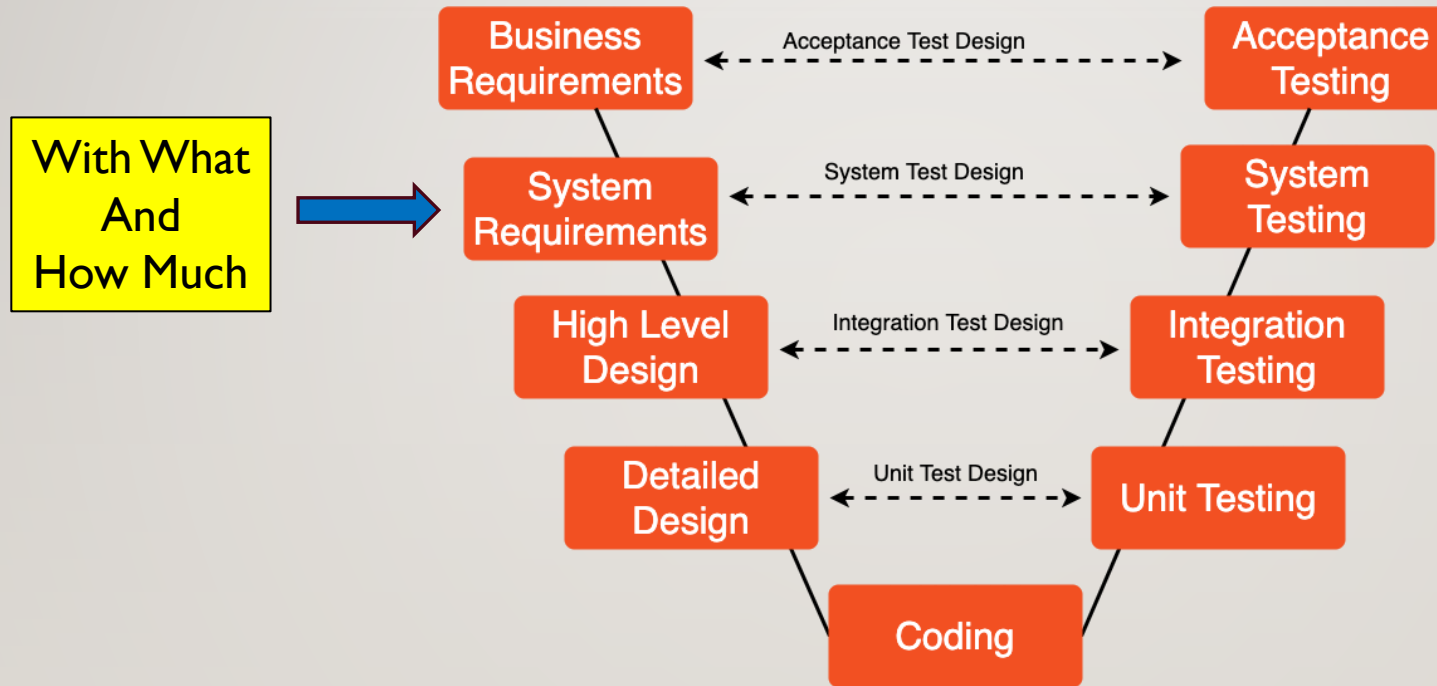
TEAM EXERCISE I

Brainstorm Business Requirement for the project and write business requirement statement

DAY 2



SW DEVELOPMENT PROCESS



SDLC - V Model - notepub.io

SYSTEM/TECHNICAL REQUIREMENT (EXAMPLE)

Technical Requirements and Metrics:

- 1. Deep Learning Models:** Train a CNN to achieve at least 98% accuracy on defect detection in validation datasets.

검증 데이터 세트에서 결함 감지에 대해 최소 98%의 정확도를 달성하도록 CNN을 훈련시킵니다.

- 2. Robotics Hardware:** Ensure the robot processes images and delivers results within 2 seconds per item, with 99.9% system uptime.

로봇이 99.9%의 시스템 가동 시간으로 항목당 2초 이내에 이미지를 처리하고 결과를 제공하도록 보장합니다.



SYSTEM/TECHNICAL REQUIREMENT (EXAMPLE)

Technical Requirements and Metrics:

- 3. Interface and Control System:** Design for less than 0.1% downtime and a response time under 1 second for user interactions.

다운타임이 0.1% 미만이고 사용자 상호 작용에 대한 응답 시간이 1초 미만으로 설계됩니다.

TEAM EXERCISE

Brainstorm System Requirement for the project and document

Using the posted notes and flipchart as needed

WHAT WE REALLY HAVE TO WORK WITH



YOUR PROJECT ENVIRONMENT



BASE HW/OS

- PC

- Ubuntu 22.04
- USB Camera



- Network
 - Wifi



- AMR

- TurtleBot4
- Ubuntu 22.04



OBJ. DET.

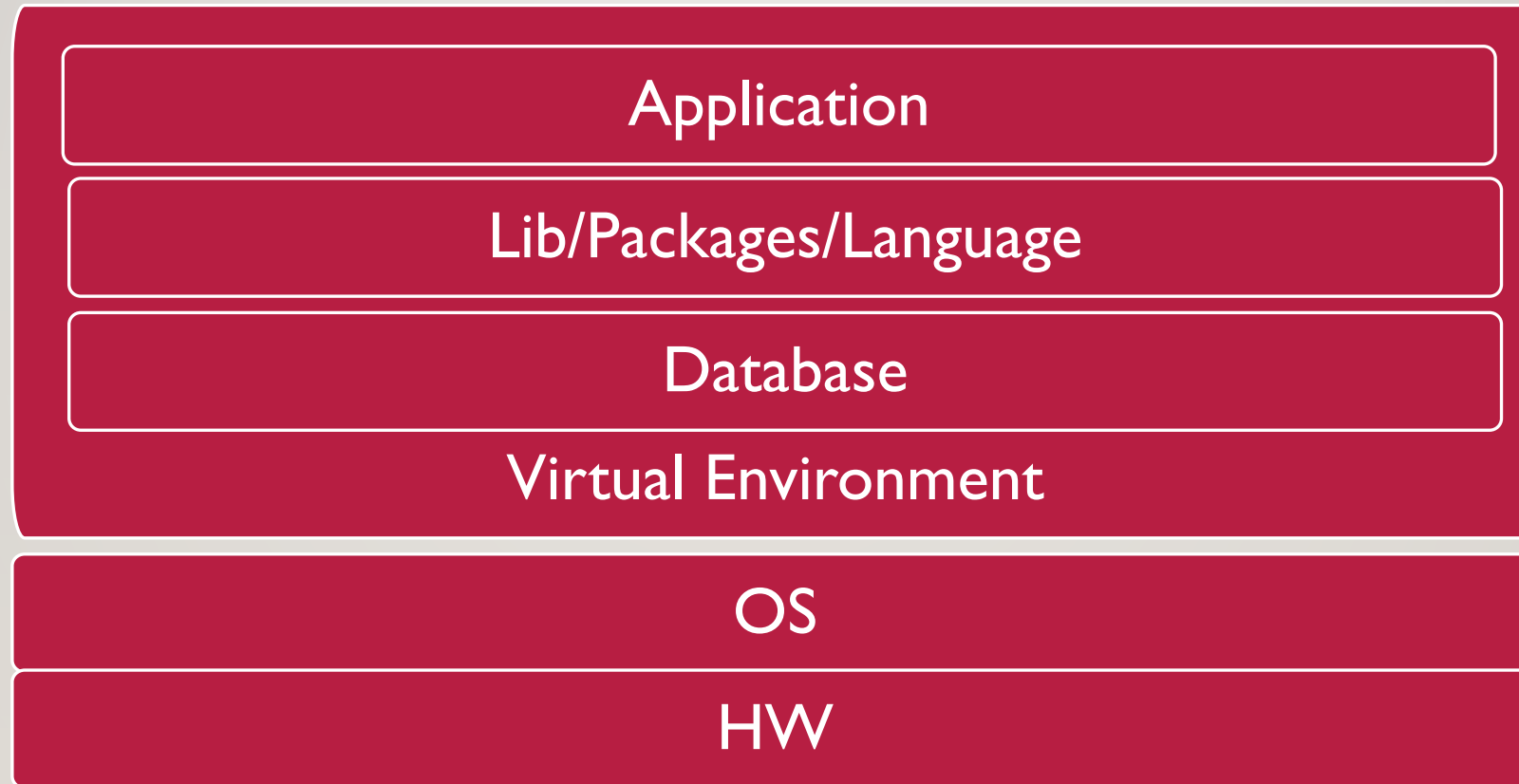
TARGET



DUMMY



EXAMPLE SYSTEM STACK



PACKAGES

PC

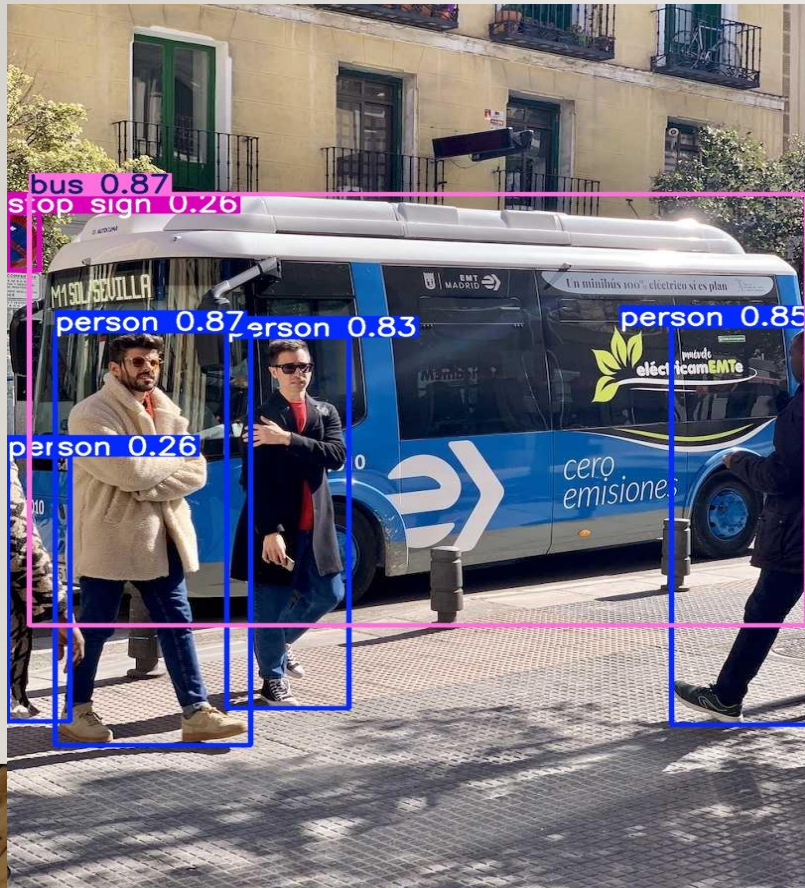
- Python3
- ROS2
- Opencv
- Ultralytics
- Flask
- SQLite3

AMR

- Python3
- ROS2
- Opencv
- Ultralytics

YOLO OBJ. DET. VS. YOLO TRACKING

- Yolo Obj. Detection



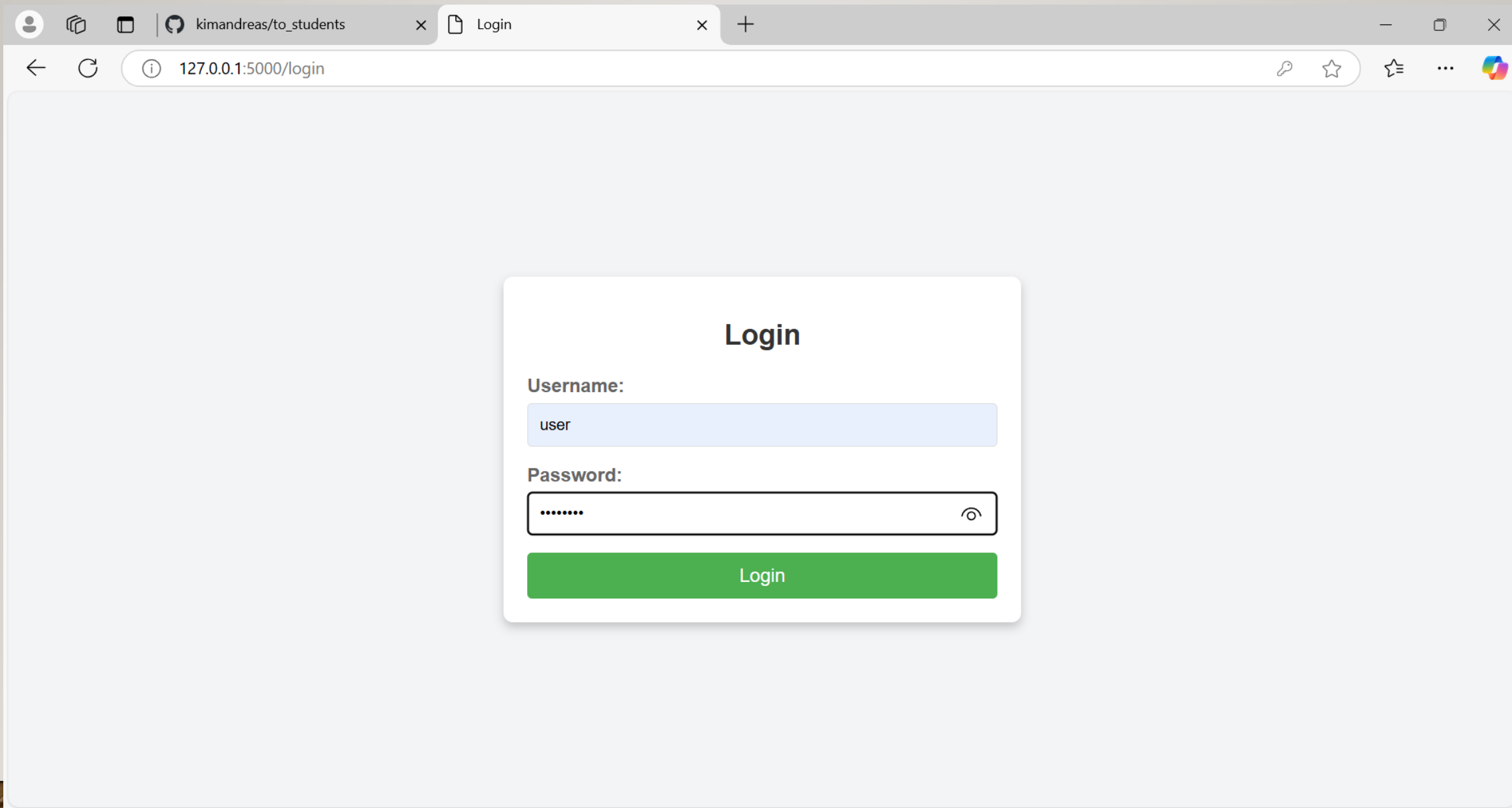
- Yolo Tracking

- [\(469\) YOLO people detection + SORT tracking – YouTube](#)
- [Bing Videos](#)
- [Track - Ultralytics YOLO Docs](#)

AMR (TURTLEBOT4)

- [Features](#) · [User Manual](#)
- <https://turtlebot.github.io/turtlebot4-user-manual/overview/features.html>






kimandreas/to_students


Welcome

127.0.0.1:5000/welcome

welcome, user!

You are now logged in.





Violations Detected

ID	Name	Date & Time
0	Truck	2024-11-06 10:30:22

Track and Following

ID	Name	Date & Time
1	Dummy	2024-11-06 10:30:22

KEY SUBSYSTEM (MODULES) TO DEVELOP

- 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 2

Brainstorm **Updated** System Requirement for the project and document

Using the posted notes and flipchart as needed

SYSTEM REQUIREMENT PRESENTATION BY EACH TEAM

Using the posted notes and flipchart as needed

EXAMPLE SYSTEM REQUIREMENT DOCUMENT

System Requirements Document (SRD)

↓

Project Title: Autonomous Mobile Robot (AMR) Security System↓

Version: 1.0↓

Date: [Insert Date]

1. Introduction

The system requirements define the technical specifications for developing and implementing the AMR-based security solution. This includes hardware, software, networking, and integration requirements.

2. System Overview

This system will provide autonomous patrolling, threat detection, and reporting for secure areas using AI-enabled Autonomous Mobile Robots (AMRs). It integrates navigation, sensor data processing, real-time alerts, and user interface management.

시스템 요구사항 문서 (SRD)

프로젝트 제목: 자율 이동 로봇(AMR) 보안 시스템↓

버전: 1.0↓

날짜: [날짜 삽입]

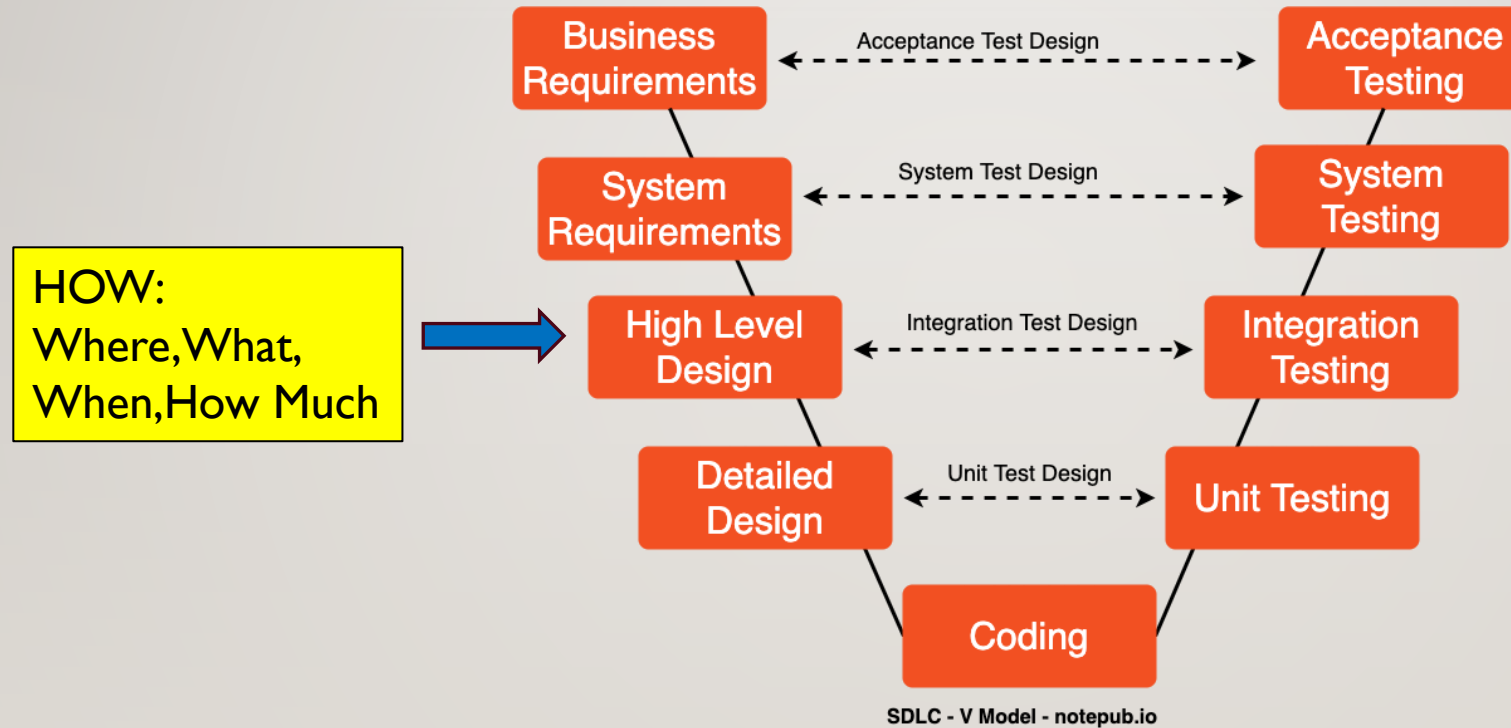
1. 소개

이 시스템 요구사항 문서는 AMR 기반 보안 솔루션의 개발 및 구현을 위한 기술 사양을 정의합니다. 여기에는 하드웨어, 소프트웨어, 네트워킹, 통합 요구사항이 포함됩니다.

2. 시스템 개요

이 시스템은 AI 기반 자율 이동 로봇(AMR)을 사용하여 보안 구역의 자율 순찰, 위험 탐지 및 보고를 제공합니다. 네비게이션, 센서 데이터 처리, 실시간 경고 및 사용자 인터페이스 관리를 통합합니다.

SW DEVELOPMENT PROCESS



OVERALL SYSTEM/HIGH LEVEL DESIGN(EXAMPLE)

- Detailed System Design with Specific Technologies and Models:

1. Deep Learning Model:

1. **Model Type:** Use a Convolutional Neural Network (CNN) based on the EfficientNet architecture for efficient and scalable image processing.

효율적이고 확장 가능한 이미지 처리를 위해 EfficientNet 아키텍처를 기반으로 하는 CNN(Convolutional Neural Network)을 사용합니다.

2. **Training:** Leverage transfer learning with pre-trained ImageNet weights as a starting point to reduce training time and improve accuracy.

사전 훈련된 ImageNet 가중치를 시작점으로 하는 전이 학습을 활용하여 훈련 시간을 단축하고 정확도를 높일 수 있습니다.

OVERALL SYSTEM DESIGN(EXAMPLE)

- Detailed System Design with Specific Technologies and Models:

2. Hardware Specifications:

1. **Robotics Arm:** Integrate a high-precision KUKA robotic arm for stable and accurate product handling.

안정적이고 정확한 제품 핸들링을 위해 고정밀 KUKA 로봇 암을 통합합니다.

2. **Cameras:** Use Sony Industrial Cameras with high frame rates and resolution to capture detailed images for defect detection.

높은 프레임 속도와 해상도의 소니 산업용 카메라를 사용하여 결함 감지를 위한 디테일한 이미지를 캡처할 수 있습니다.



OVERALL SYSTEM DESIGN(EXAMPLE)

- Detailed System Design with Specific Technologies and Models:

3. Software Technologies:

1. **Framework:** Develop the deep learning model using TensorFlow and Keras for their extensive support and community.

광범위한 지원과 커뮤니티를 위해 TensorFlow 및 Keras를 사용하여 딥 러닝 모델을 개발하세요.

2. **Server Technology:** Utilize NVIDIA DGX systems for high-throughput and low-latency processing, crucial for real-time applications.

실시간 애플리케이션에 중요한 높은 처리량과 짧은 대기 시간 처리를 위해 NVIDIA DGX 시스템을 활용하세요.

OVERALL SYSTEM DESIGN(EXAMPLE)

- Detailed System Design with Specific Technologies and Models:

4. Interface and Control System:

1. **Dashboard:** Build the user interface using React.js for its efficient rendering performance, supported by a Node.js backend for handling API requests.

효율적인 렌더링 성능을 위해 React.js를 사용하여 사용자 인터페이스를 구축하고, API 요청을 처리하기 위한 Node.js 백엔드에서 지원합니다.

2. **Communication:** Implement MQTT for lightweight, real-time messaging between the robotic components and the server.

로봇 구성 요소와 서버 간의 경량 실시간 메시징을 위해 MQTT를 구현합니다.

OVERALL SYSTEM DESIGN(EXAMPLE)

- Detailed System Design with Specific Technologies and Models:

5. Integration and Testing Techniques:

1. **Simulation Software:** Use ROS (Robot Operating System) for simulation and to prototype interactions between components.

ROS(Robot Operating System)를 사용하여 구성요소 간의 상호 작용을 시뮬레이션하고 프로토타이핑할 수 있습니다.

2. **Automated Testing:** Integrate Jenkins for continuous integration, ensuring every code commit is built, tested, and errors are addressed promptly.

지속적인 통합을 위해 Jenkins를 통합하여 모든 코드 커밋을 빌드, 테스트하고 오류를 신속하게 해결할 수 있습니다.

BASE HW/OS

- PC

- Ubuntu 22.04
- USB Camera



- Network
 - Wifi



- AMR

- TurtleBot4
- Ubuntu 22.04



OBJ. DET.

TARGET



DUMMY

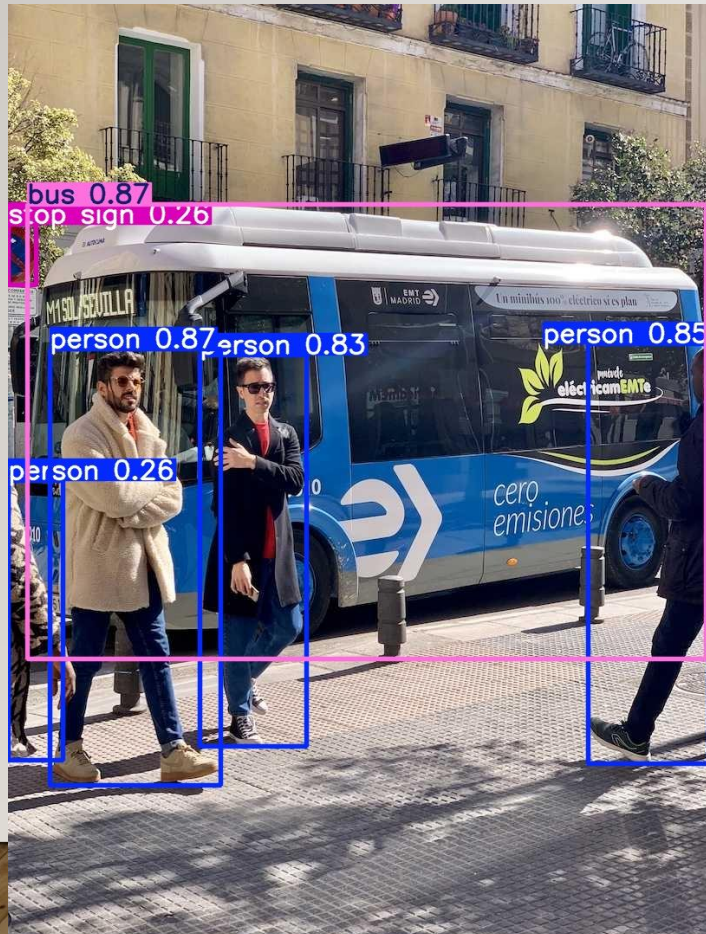


AMR (DEMO)

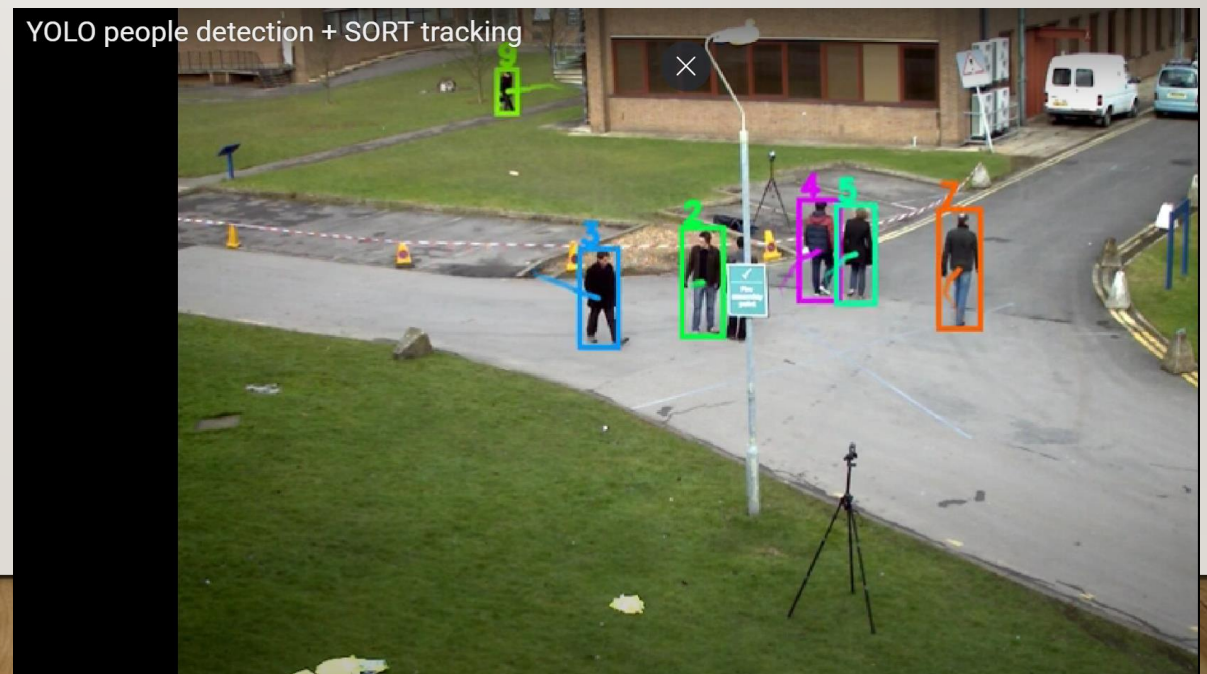
- Power On and Off
- Connecting to Robot
 - Wifi Router
 - SSH
- Docking and Undocking
- Teleop with keyboard
- Navigation with SLAM

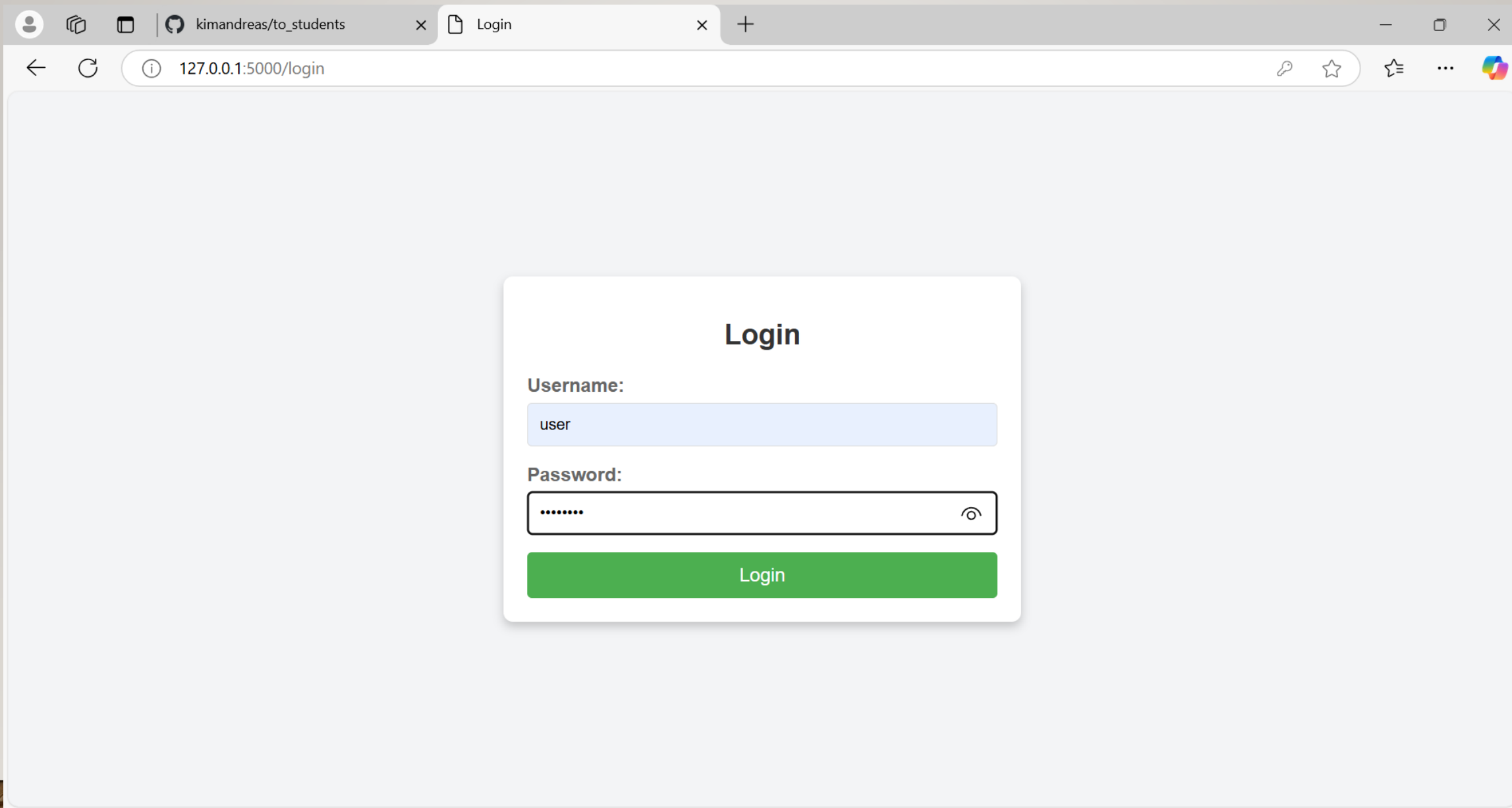


YOLO OBJ. DET. VS. YOLO TRACKING



- [Track - Ultralytics YOLO Docs](#)
 - [\(469\) YOLO people detection + SORT tracking – YouTube](#)
 - [Bing Videos](#)





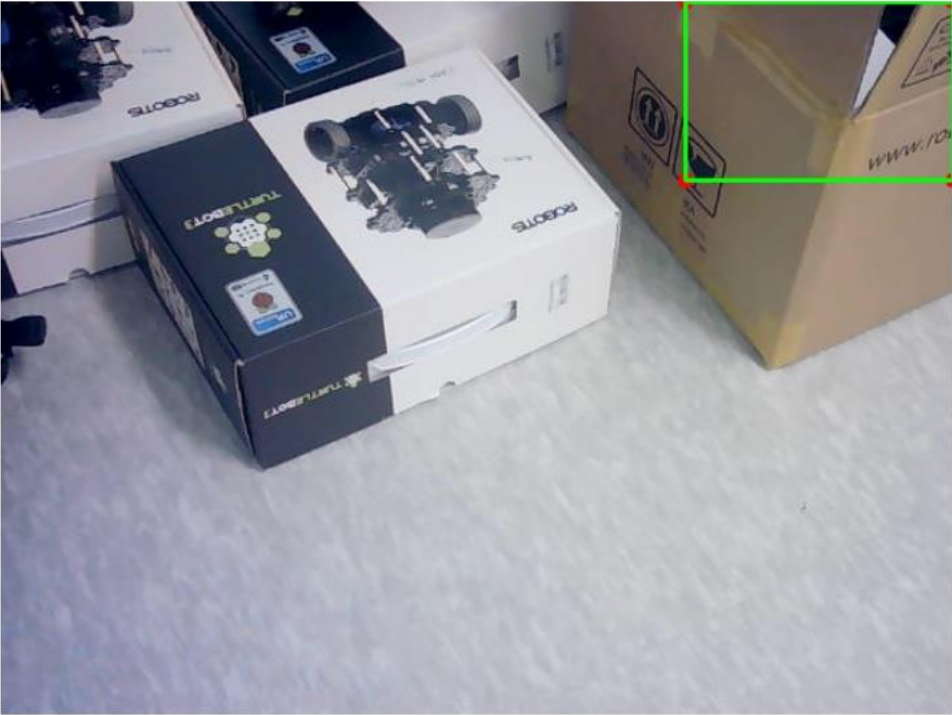
kimandreas/to_students

Welcome

127.0.0.1:5000/welcome


welcome, user!

You are now logged in.



Violations Detected

ID	Name	Date & Time
0	Truck	2024-11-06 10:30:22



Track and Following

ID	Name	Date & Time
1	Dummy	2024-11-06 10:30:22

KEY SUBSYSTEM (MODULES) TO DEVELOP

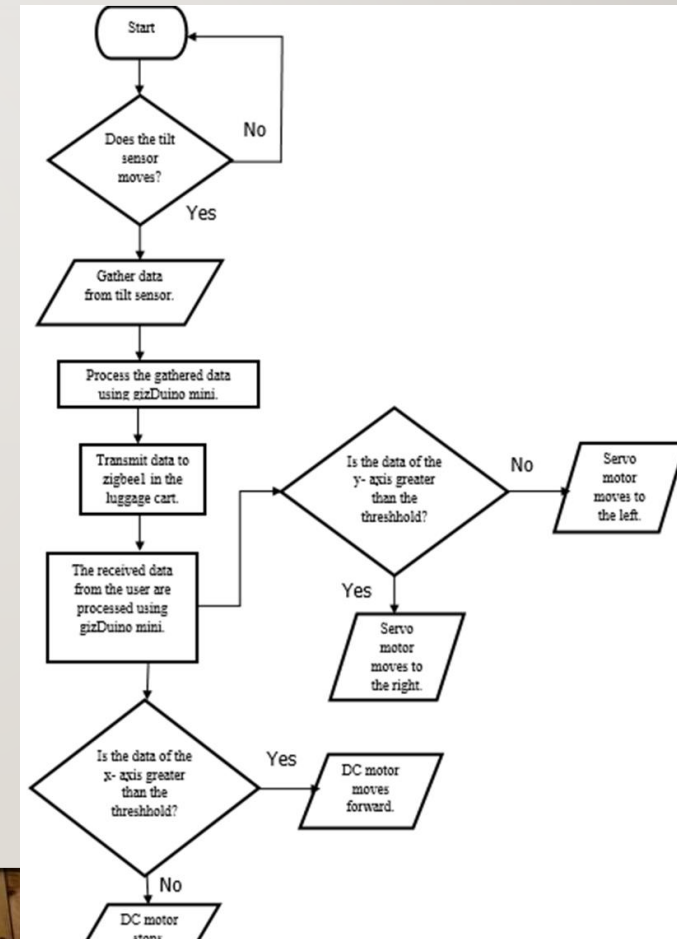
- Detection Alert
 - Camera Capture
 - Object Detection
 - Send messages to other subsystems
- AMR Controller
 - Receive messages and act accordingly
 - Move using (SLAM) with Obstruction avoidance
 - Target Acquisition (Obj. Det.) and Tracking
 - Follow target using camera and motor control
- System Monitor (OPTION)
 - Receive and Display Detection Camera and info
 - Receive and Display AMR Camera and info
 - Store, display, and report Information and Alerts

VISUALIZATION – SYSTEM FUNCTIONAL PROCESS FLOW DIAGRAMS

- To-Be Functional Process Flow Diagram

Detection Alert
System Monitor
AMR Controller

- F**unctions
- I**nterfaces
Dataflow
- T**esting (Coding & Testing Plan)
Error and Exception Handling



TEAM EXERCISE 3

Create System Design using Process Flow Diagram.

Use the posted notes and flipchart as needed

SYSTEM DESIGN PRESENTATION BY EACH TEAM



EXAMPLE SYSTEM DESIGN DOCUMENT

System Design Document (SDD)❧

Project Title: Autonomous Mobile Robot (AMR) Security System↓

Version: 1.1↓

Date: [Insert Date]❧

1. Overview❧

The Autonomous Mobile Robot (AMR) Security System is designed to provide autonomous patrolling, threat detection, and alerting within a secure area using a single AI-enabled robot. The system consists of one AMR equipped with necessary hardware and software components to operate independently, processing data on-board without the need for a central server.❧

2. System Architecture❧

Since the system consists of a single AMR, data processing, navigation, threat detection, and alerting are all performed locally on the AMR itself. The AMR communicates directly with a user interface on a PC via a local network (Wi-Fi) for monitoring, alerts, and manual override if required.❧

시스템 설계 문서 (SDD)❧

프로젝트 제목: 자율 이동 로봇(AMR) 보안 시스템↓

버전: 1.1↓

날짜: [날짜 삽입]❧

1. 개요❧

자율 이동 로봇(AMR) 보안 시스템은 단일 AI 기반 로봇을 사용하여 보안 구역 내에서 자율 순찰, 위협 탐지 및 경고를 제공하도록 설계되었습니다. 시스템은 단일 AMR이 독립적으로 작동할 수 있도록 필요한 하드웨어 및 소프트웨어 구성 요소로 구성되며, 중앙 서버 없이 데이터를 현장에서 처리합니다.❧

2. 시스템 아키텍처❧

이 시스템은 단일 AMR으로 구성되므로 데이터 처리, 네비게이션, 위협 탐지 및 경고가 모두 AMR에서 로컬로 수행됩니다. AMR은 모니터링, 알림 및 수동 제어를 위해 PC의 사용자 인터페이스와 로컬 네트워크(Wi-Fi)를 통해 직접 통신합니다.❧

PROJECT TIMELINE/CRITICAL PATH ITEM MANAGEMENT



EX. IMPLEMENTATION TIMELINE

Function Backlog	Owner	5월 20일	5월 21일	5월 22일	5월 23일	5월 24일	5월 25일
Unloading Module	John						
Input1	John						
Input2	John						
Output 1	John						
Unit Test	John						
Receiving Module	Jan						
Input1	Feb						
Input2	Mar						
Output 1	Apr						
Unit Test	John						
Integration Test	John/Jan						

이 타임라인을 생성할 때
먼저 시스템 및 시스템
설계의 기능 프로세스
다이어그램(To-Be)을
완료해야 합니다.

그런 다음 각 기능(하위
함수/모듈 및
입력/출력)에 대해 누가,
무엇을, 언제, 어떻게
정의합니다. 표에 설명
타임라인 형식의 무엇을,
누가, 언제를 입력합니다.

CRITICAL PATH ITEMS LIST

- tasks that directly impact the project timeline. Delays in these tasks would delay the project's overall completion because they represent the longest stretch of dependent activities
- 프로젝트 타임라인에 직접적인 영향을 주는 작업입니다. 이러한 작업이 지연되면 종속 활동이 가장 길어지기 때문에 프로젝트의 전체 완료가 지연됩니다

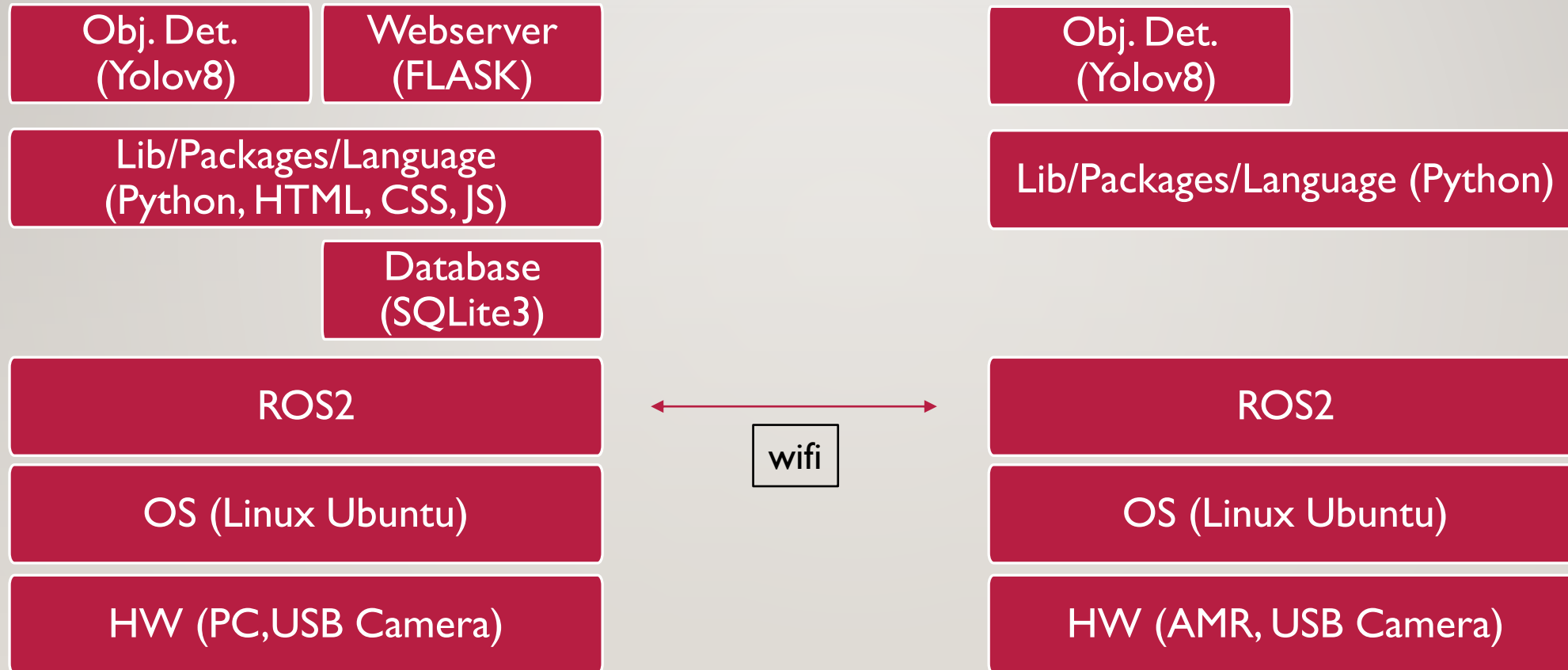
GUIDE TO PROGRESS INDICATORS

- Project Timeline
 - **Green** – less 10% of the listed items are delayed
 - **Yellow** – more than 10% but less than 20% of listed items are delayed
 - **Red** – more than 20% of listed items are delayed
- Critical Path Items
 - **Green** – reduced number of item(s)
 - **Yellow** – no new item(s)
 - **Red** – additional item(s)

SYSTEM AND DEVELOPMENT ENVIRONMENT SETUP



PROJECT SW STACK



USEFUL COMMANDS

\$ lsb_release -a

- Linux distribution info

\$ echo \$ROS_DISTRO

- ROS: Humble

\$ code --version

- Vscode

\$ python3 --version

- Python

\$ sudo apt update

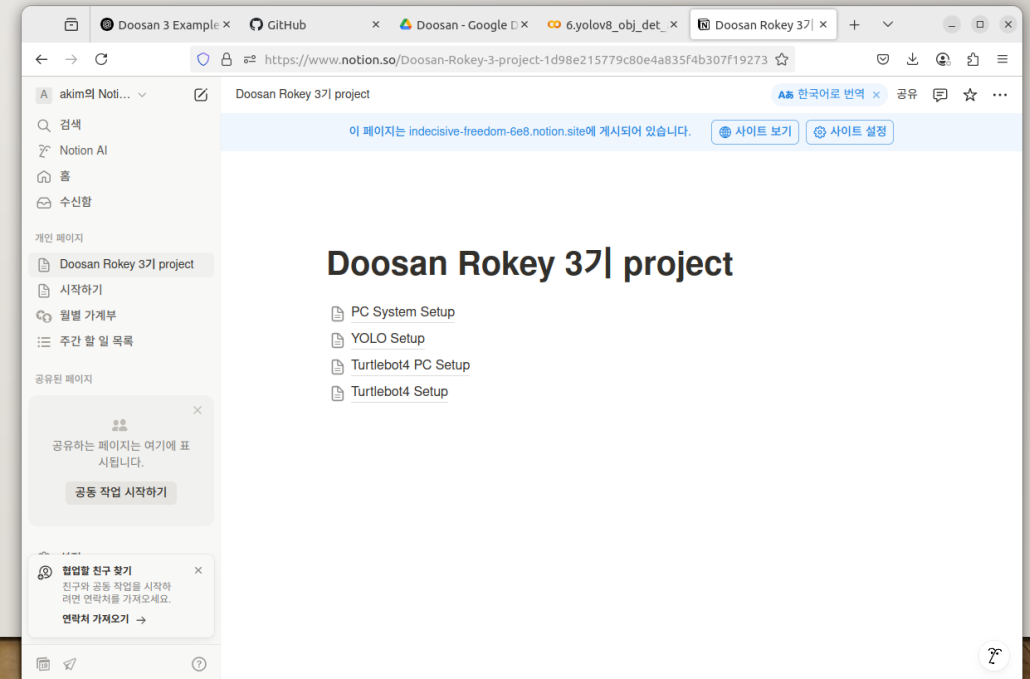
\$ sudo apt upgrade

\$ python -m ensurepip --upgrade

- Assumes Linux (Ubuntu 22.04), ROS Humble, VScode, and Python are already installed globally

SYSTEM ENVIRONMENT SETUP SHELL SCRIPT

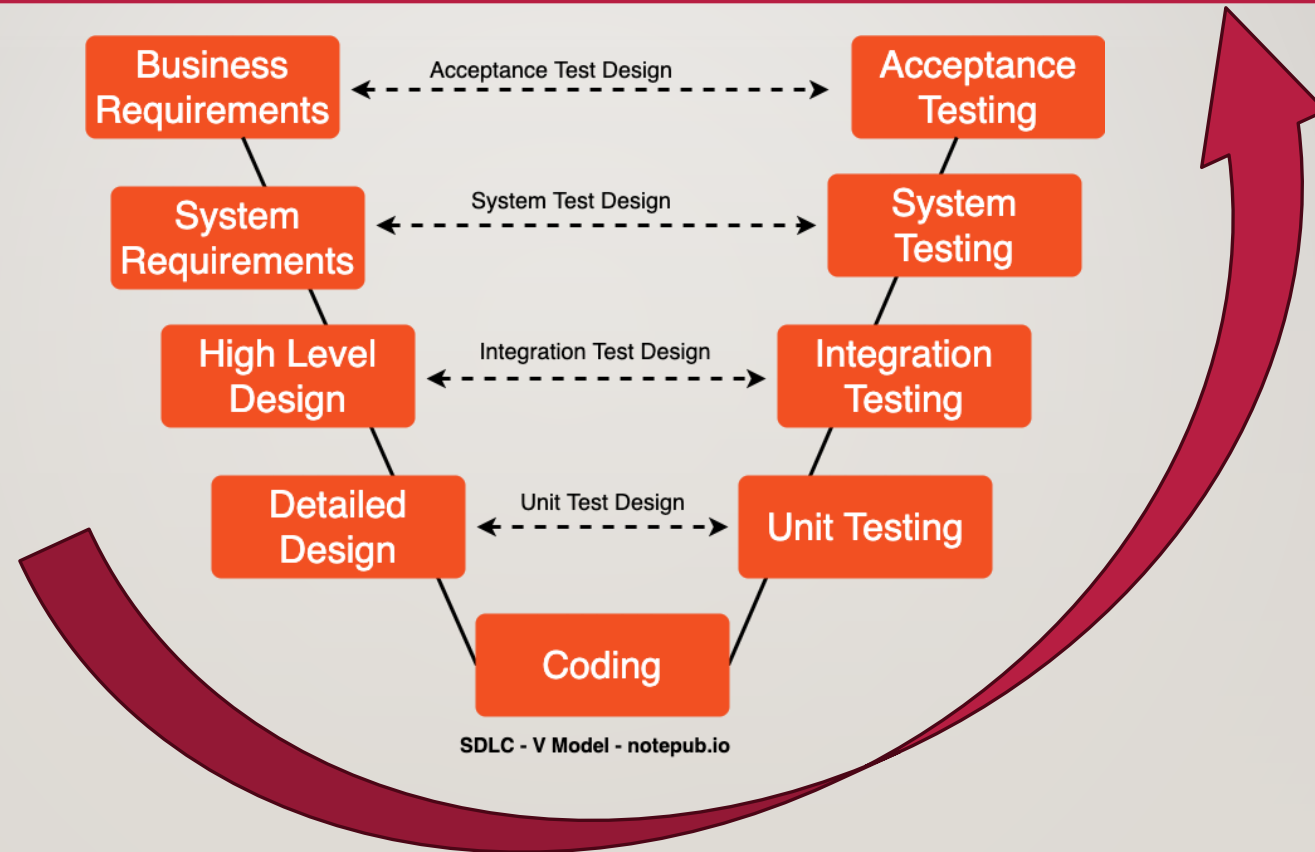
- PC System Setup
- <https://indecisive-freedom-6e8.notion.site/PC-System-Setup-1d98e215779c806080bbd1014d63a406>



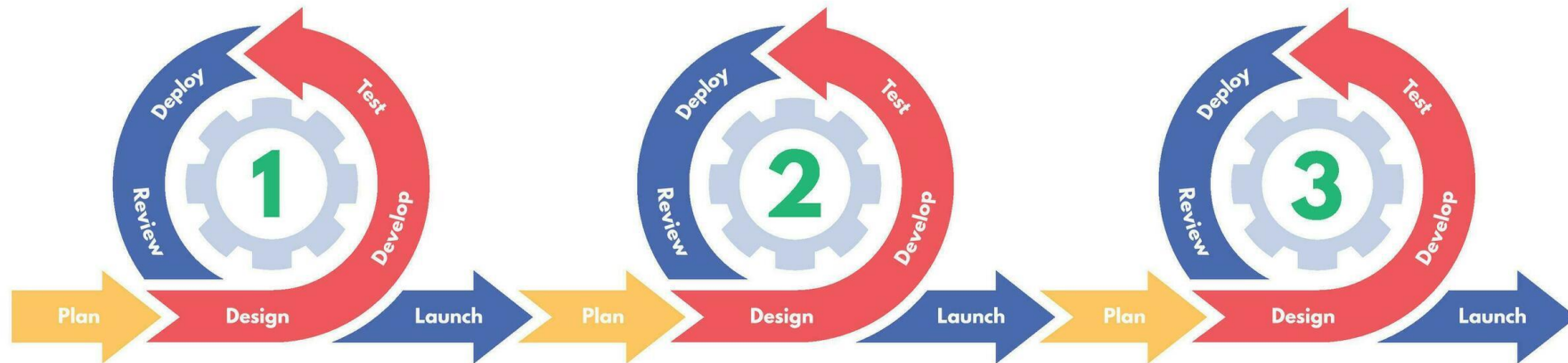
DETAIL DESIGN TO USER ACCEPTANCE



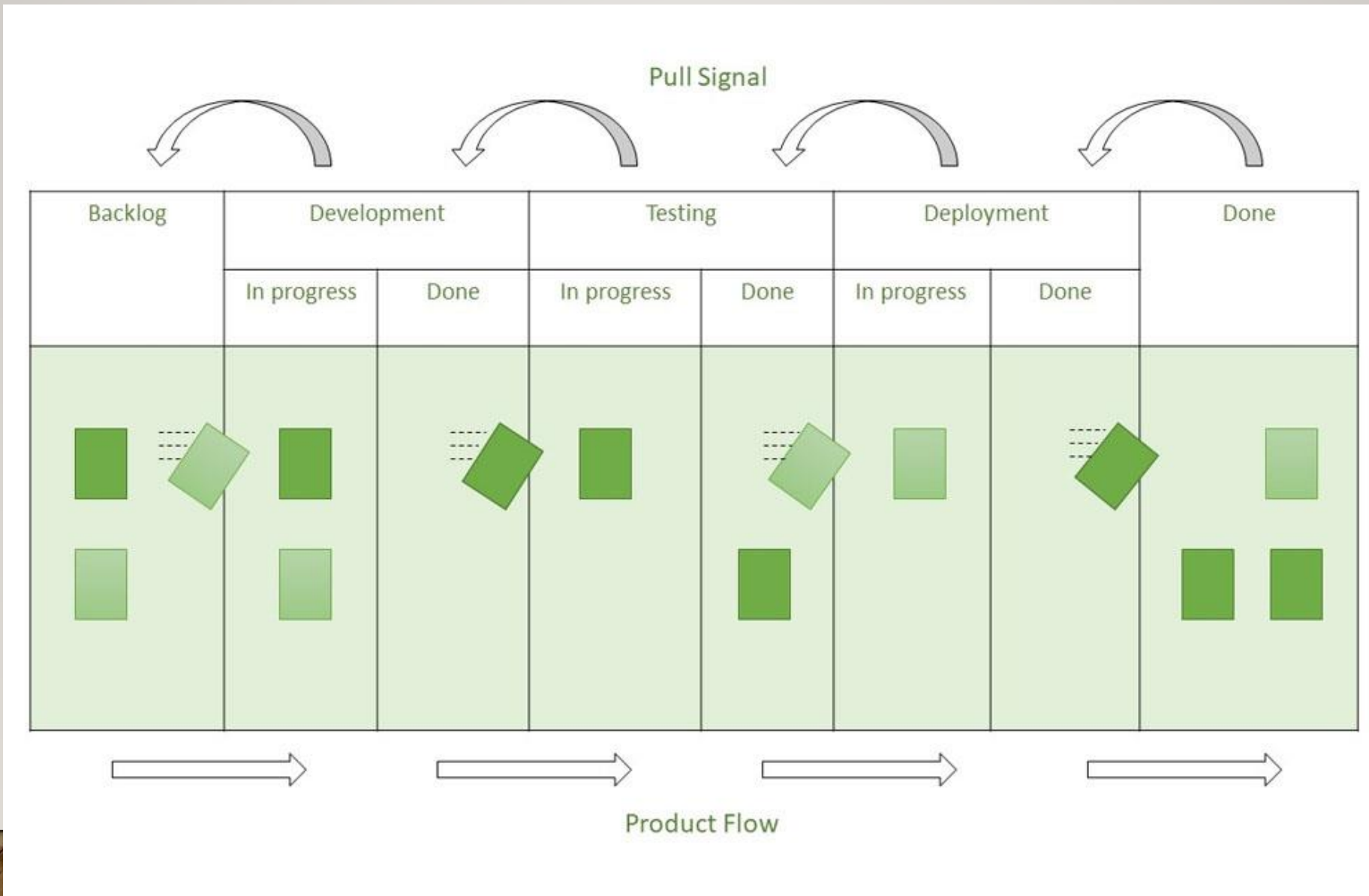
SW DEVELOPMENT PROCESS



AGILE DEVELOPMENT

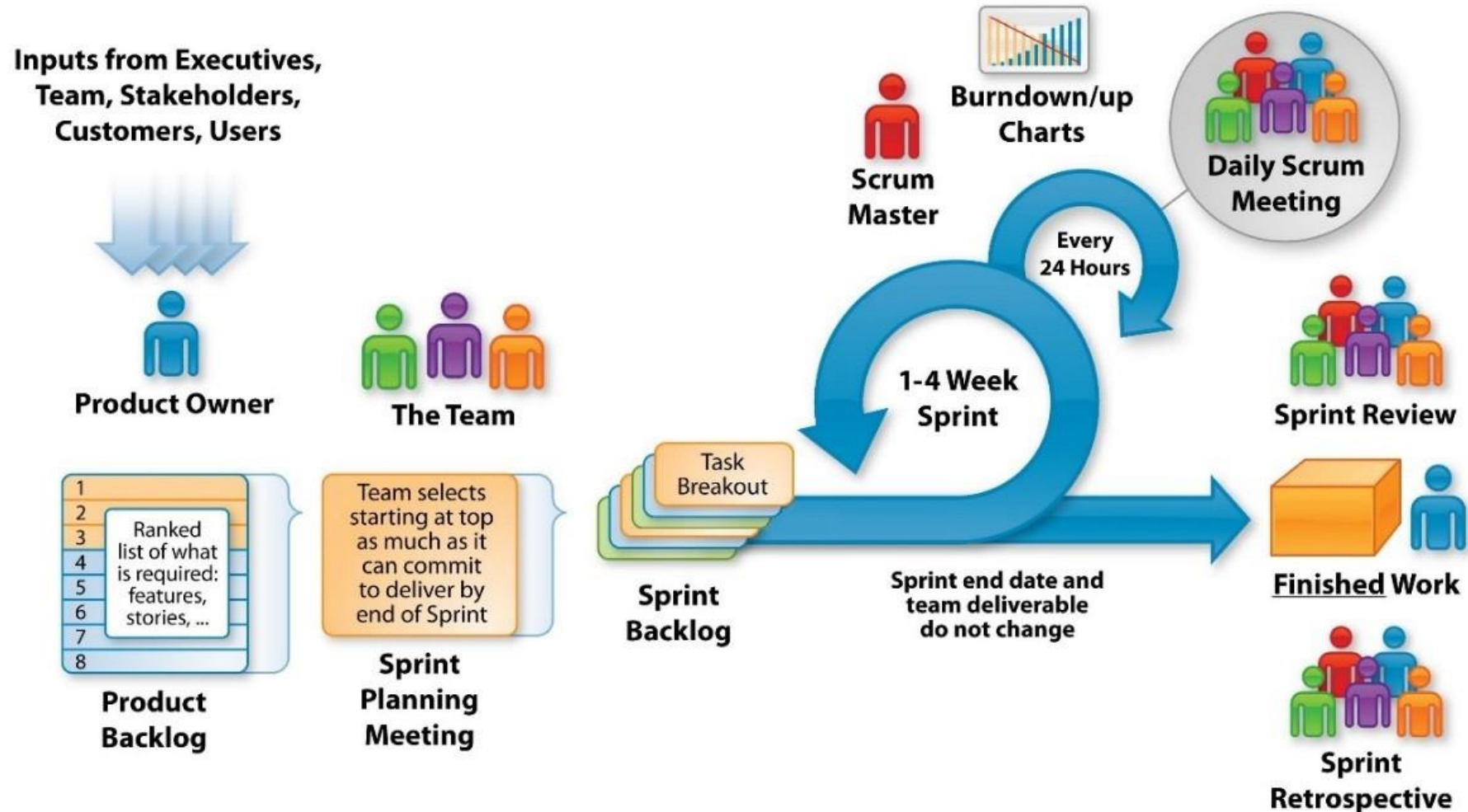


KANBAN METHODOLOGY





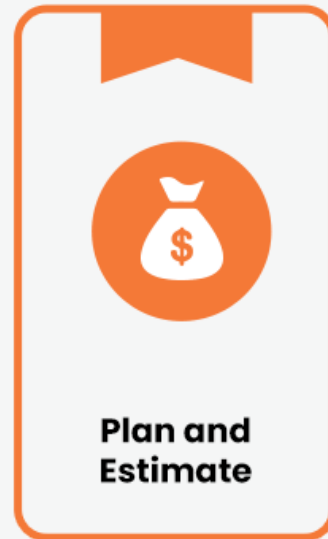
The Agile - Scrum Framework



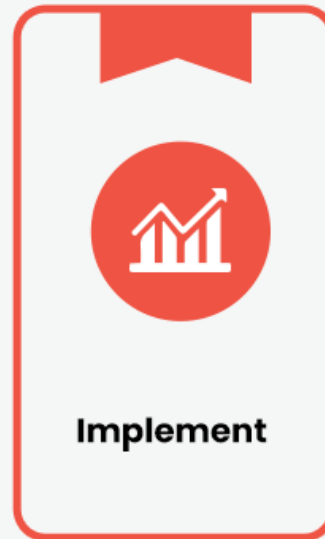
5 Stages of Scrum Sprint



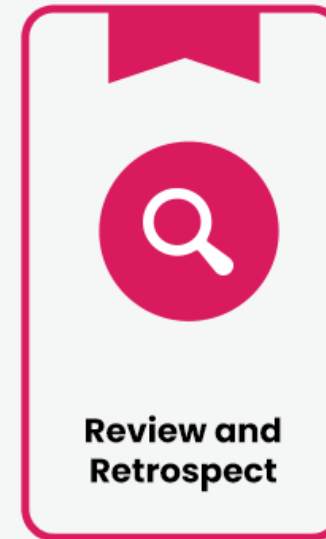
This phase includes the processes related to the commencement of a project, such as a scope and objectives, creating and distributing its charter, and taking other steps to guarantee success.



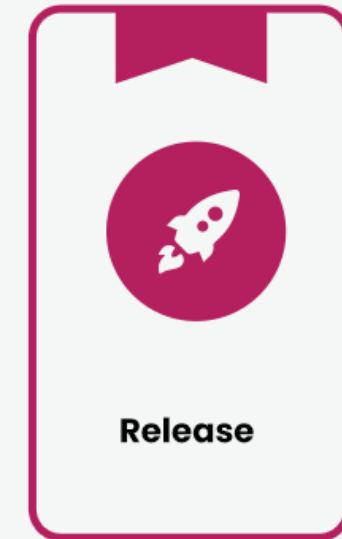
This phase involves planning and estimating processes, including creating user stories, approving, assessing, committing user stories, creating tasks, evaluating tasks, and creating a Sprint backlog.



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.

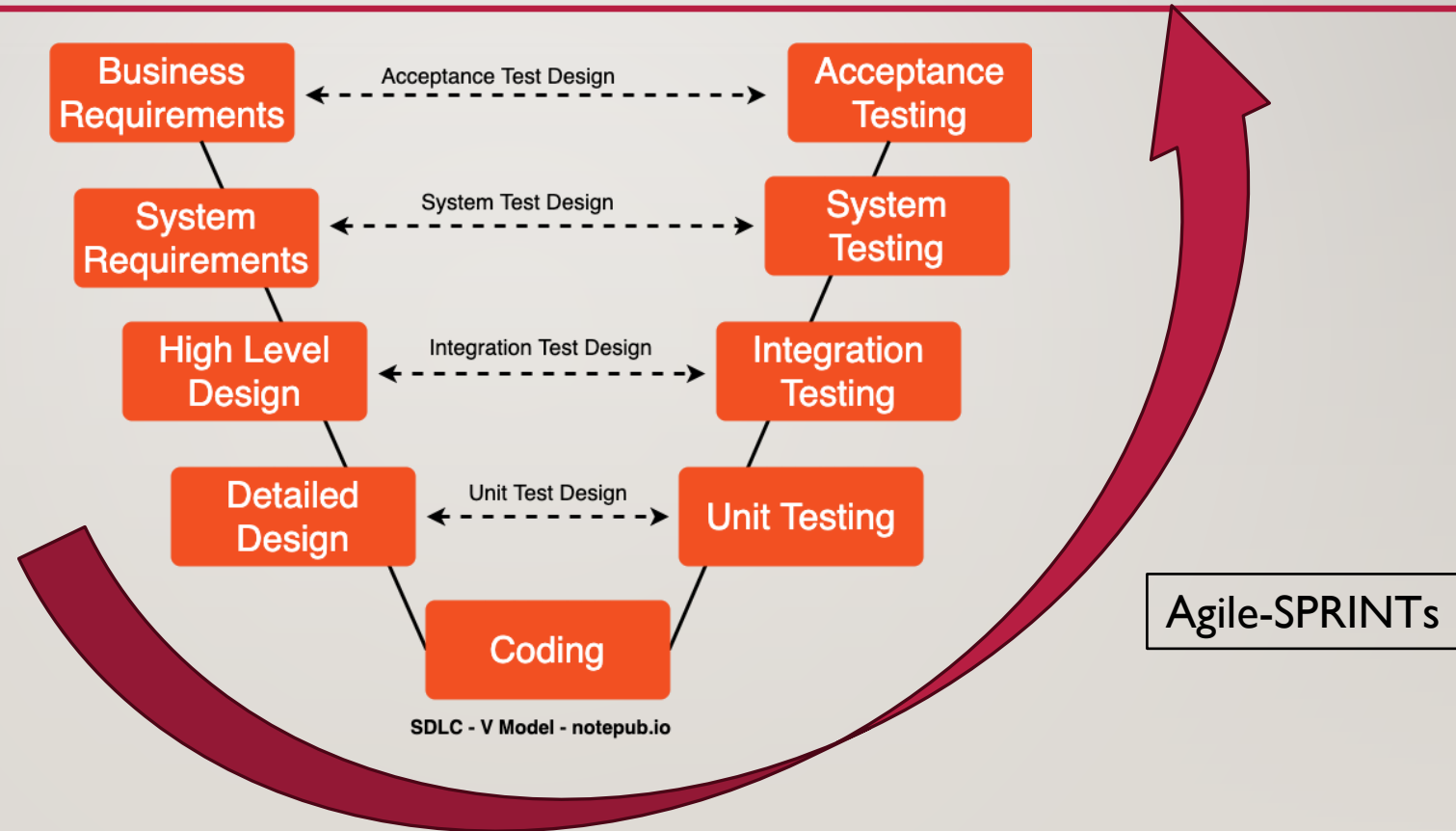


This stage of the project lifecycle is concerned with evaluating what has been accomplished so far, whether the team has worked to plan, and how it can do things better in the future.



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

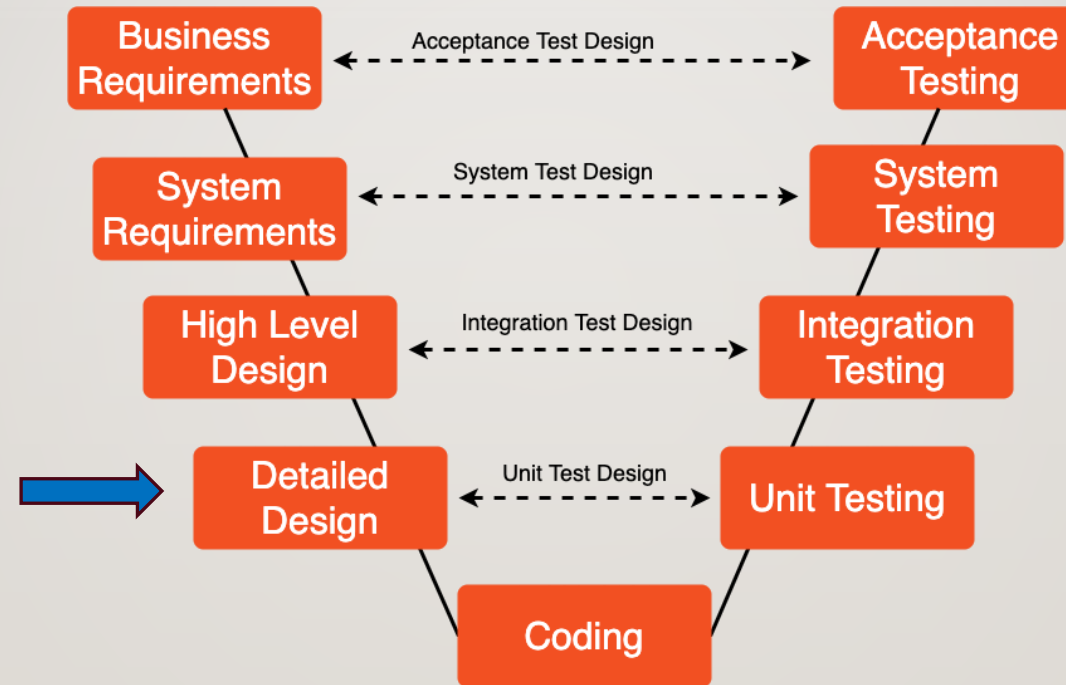
- System Monitor

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

DETECTION ALERT SPRINT

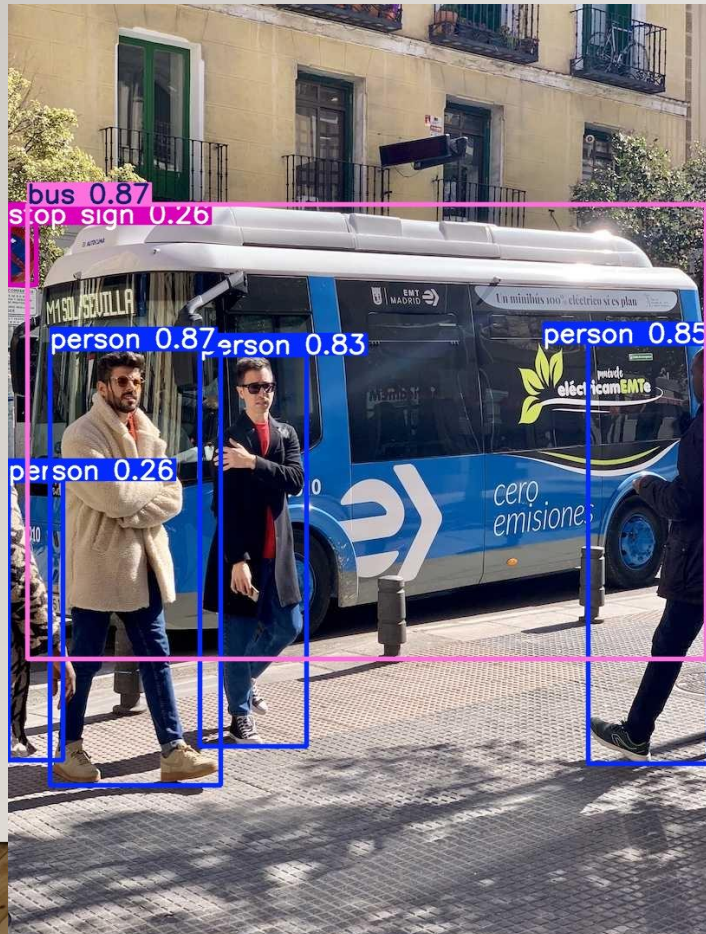


SPRINT I - DETECTION ALERT

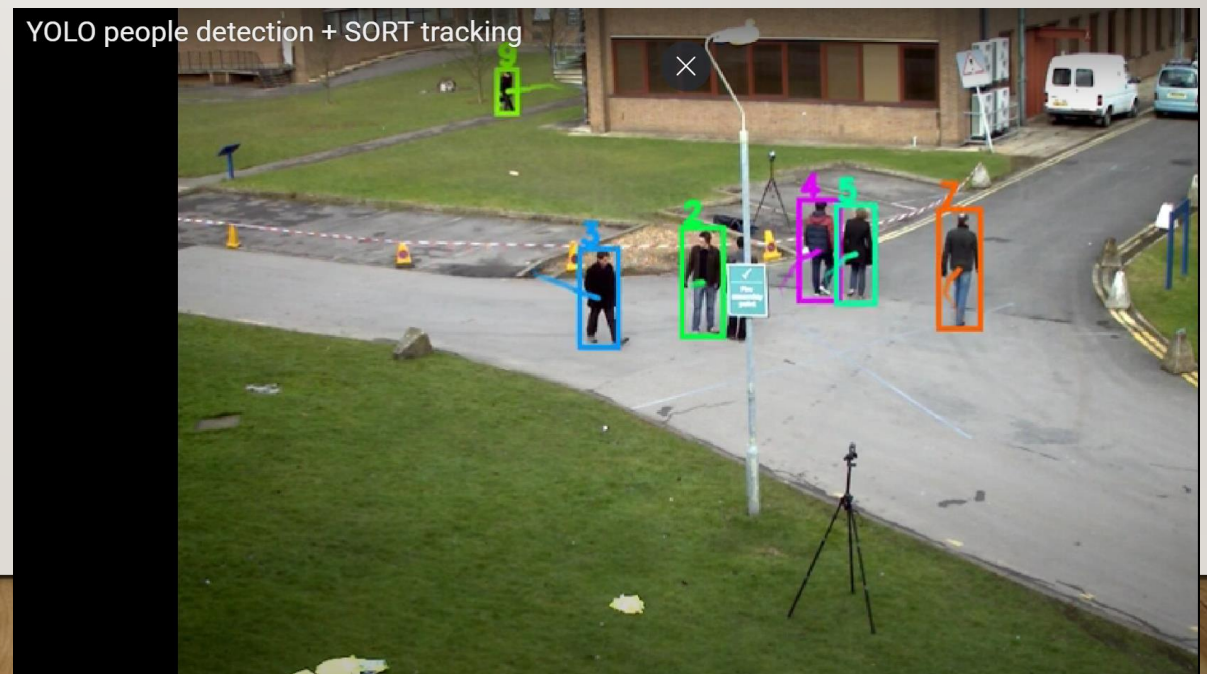


SDLC - V Model - notepub.io

YOLO OBJ. DET. VS. YOLO TRACKING



- [Track - Ultralytics YOLO Docs](#)
 - [\(469\) YOLO people detection + SORT tracking – YouTube](#)
 - [Bing Videos](#)



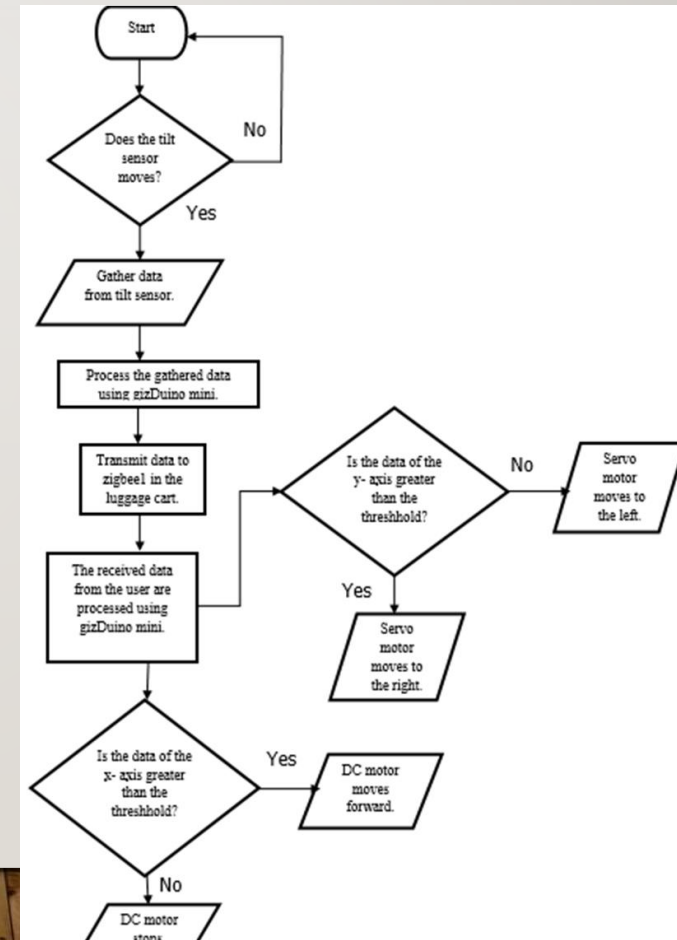
VISUALIZATION – DETAILED FUNCTIONAL PROCESS DIAGRAMS

- To-Be Functional Process Diagram

Detection Alert

PC or AMR or Both

Remember **F.I.T!!!**



TEAM EXERCISE 4

Perform Detail Design of Detection Alert Module using Process Flow Diagram

DETAIL DESIGN REVIEW BY EACH TEAM

Using the process flow diagram present team's design

EXAMPLE DETAILED DESIGN DOCUMENT

Detailed Design Document: AMR Navigation and Threat Detection

Project Title: Autonomous Mobile Robot (AMR) Security System

Version: 1.0

Date: [Insert Date]

1. Overview

This document outlines the detailed design for the Autonomous Mobile Robot (AMR) navigation and threat detection components. It covers the architecture, algorithms, data processing, and system interactions necessary to enable autonomous navigation within a secure area and real-time threat detection using onboard sensors.

2. System Architecture

The AMR system relies on onboard hardware (e.g., sensors, cameras, Jetson-Orin processor) and software (ROS2, OpenCV, YOLO) for autonomous navigation and real-time threat detection. All processing occurs locally on the AMR, with the capability to transmit alerts to a monitoring PC via Wi-Fi.

상세 설계 문서: AMR 네비게이션 및 위협 탐지

프로젝트 제목: 자율 이동 로봇(AMR) 보안 시스템

버전: 1.0

날짜: [날짜 삽입]

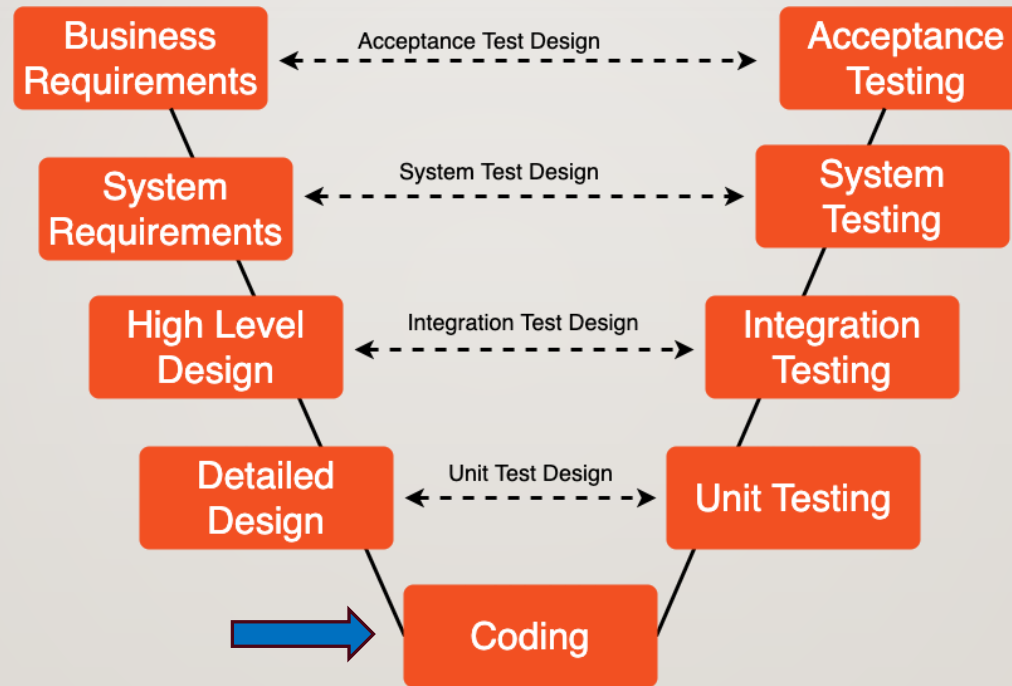
1. 개요

이 문서는 자율 이동 로봇(AMR)의 네비게이션 및 위협 탐지 구성 요소에 대한 상세 설계를 다룹니다. 자율 네비게이션과 실시간 위협 탐지를 위해 온보드 센서를 사용하는 데 필요한 아키텍처, 알고리즘, 데이터 처리 및 시스템 상호작용이 포함되어 있습니다.

2. 시스템 아키텍처

AMR 시스템은 자율 네비게이션 및 실시간 위협 탐지를 위해 온보드 하드웨어(예: 센서, 카메라, Jetson-Orin 프로세서)와 소프트웨어(ROS2, OpenCV, YOLO)를 활용합니다. 모든 처리는 AMR 내에서 로컬로 수행되며, 잠재적인 위협이 감지되면 Wi-Fi를 통해 모니터링 PC로 알림을 전송할 수 있습니다.

SPRINT I - DETECTION ALERT



SDLC - V Model - notepub.io

PERFORM DATA COLLECTION FOR DETECTION ALERT



CODING HINTS

- Image Capture

▼ day2

🔗 __init__.py

🔗 2_1_a_capture_wc_image.py

🔗 2_1_b_cont_capture_wc_image.py

🔗 2_1_c_capture_wc_thread.py

CODING HINTS

- Image Capture

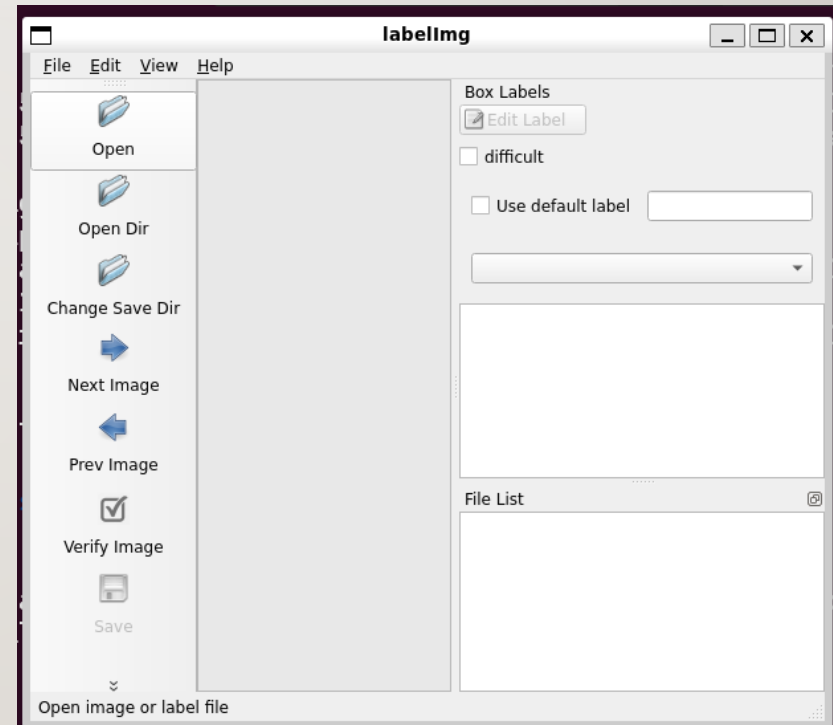
- Data Labelling

- labellmg

```
▼ day2
  ├── __init__.py
  ├── 2_1_a_capture_wc_image.py
  ├── 2_1_b_cont_capture_wc_image.py
  └── 2_1_c_capture_wc_thread.py
```

CODING HINTS

- Data Labelling : use previously installed Labellmg
- Or
- pip3 install labellmg
 - May also need “pip3 install PyQt5 lxml”
- labellmg



CODING HINTS

- Data Labelling : Labellmg

라벨링 순서

1. 이미지파일 불러오기 (Open Dir)
2. 저장형식 변경 (PascalVOC, YOLO)
3. 이미지 선택
4. 바운딩 박스 그리기(create rectbox)
5. Class 지정
6. 저장경로 생성 및 변경(Change Save Dir)
7. 저장(Save)

단축키

Ctrl + u	Load all of the images from a directory
Ctrl + r	Change the default annotation target dir
Ctrl + s	Save
Ctrl + d	Copy the current label and rect box
Ctrl + Shift + d	Delete the current image
Space	Flag the current image as verified
w	Create a rect box
d	Next image
a	Previous image
del	Delete the selected rect box
Ctrl++	Zoom in
Ctrl--	Zoom out
↑→↓←	Keyboard arrows to move selected rect box

LABELIMG ERROR CORRECTION

```
labelimg: command not found
❗ rokey-kim:~/Documents/ros2_ws$ labelImg
Traceback (most recent call last):
  File "/home/rokey-kim/.local/lib/python3.10/site-packages/libs/canvas.py", line 530, in paintEvent
    p.drawLine(self.prev_point.x(), 0, self.prev_point.x(), self.pixmap.height())
TypeError: arguments did not match any overloaded call:
  drawLine(self, QLineF): argument 1 has unexpected type 'float'
  drawLine(self, QLine): argument 1 has unexpected type 'float'
  drawLine(self, int, int, int, int): argument 1 has unexpected type 'float'
  drawLine(self, QPoint, QPoint): argument 1 has unexpected type 'float'
  drawLine(self, Union[QPointF, QPoint], Union[QPointF, QPoint]): argument 1 has unexpected type 'float'
QBackingStore::endPaint() called with active painter; did you forget to destroy it or call QPainter::end() on it?
QPainter::begin: Painter already active
Segmentation fault (core dumped)
❗ rokey-kim:~/Documents/ros2_ws$ labelImg
Traceback (most recent call last):
  File "/home/rokey-kim/.local/lib/python3.10/site-packages/libs/canvas.py", line 531, in paintEvent
    p.drawLine(0, self.prev_point.y(), self.pixmap.width(), self.prev_point.y())
TypeError: arguments did not match any overloaded call:
  drawLine(self, QLineF): argument 1 has unexpected type 'int'
  drawLine(self, QLine): argument 1 has unexpected type 'int'
  drawLine(self, int, int, int, int): argument 2 has unexpected type 'float'
  drawLine(self, QPoint, QPoint): argument 1 has unexpected type 'int'
  drawLine(self, Union[QPointF, QPoint], Union[QPointF, QPoint]): argument 1 has unexpected type 'int'
QBackingStore::endPaint() called with active painter; did you forget to destroy it or call QPainter::end() on it?
QPainter::begin: Painter already active
Segmentation fault (core dumped)
```

LABELIMG ERROR CORRECTION

```
Labelimg: command not found
rokey-kim:~/Documents/ros2_ws$ labelimg
Traceback (most recent call last):
  File "/home/rokey-kim/.local/lib/python3.10/site-packages/libs/canvas.py", line 530, in paintEvent
    p.drawLine(self.prev_point.x(), 0, self.prev_point.x(), self.pixmap.height())
TypeError: arguments did not match any overloaded call:
  drawLine(self, QLineF): argument 1 has unexpected type 'float'
  drawLine(self, QLine): argument 1 has unexpected type 'float'
  drawLine(self, int, int, int, int): argument 1 has unexpected type 'float'
  drawLine(self, QPoint, QPoint): argument 1 has unexpected type 'float'
  drawLine(self, Union[QPointF, QPoint], Union[QPointF, QPoint]): argument 1 has unexpected type 'float'
QBackingStore::endPaint() called with active painter; did you forget to destroy it or call QPainter::end() on it?
QPainter::begin: Painter already active
Segmentation fault (core dumped)
rokey-kim:~/Documents/ros2_ws$ labelimg
Traceback (most recent call last):
  File "/home/rokey-kim/.local/lib/python3.10/site-packages/libs/canvas.py", line 531, in paintEvent
    p.drawLine(0, self.prev_point.y(), self.pixmap.width(), self.prev_point.y())
TypeError: arguments did not match any overloaded call:
  drawLine(self, QLineF): argument 1 has unexpected type 'int'
  drawLine(self, QLine): argument 1 has unexpected type 'int'
  drawLine(self, int, int, int, int): argument 2 has unexpected type 'float'
  drawLine(self, QPoint, QPoint): argument 1 has unexpected type 'int'
  drawLine(self, Union[QPointF, QPoint], Union[QPointF, QPoint]): argument 1 has unexpected type 'int'
QBackingStore::endPaint() called with active painter; did you forget to destroy it or call QPainter::end() on it?
QPainter::begin: Painter already active
Segmentation fault (core dumped)
```

```
525         p.setBrush(brush)
526         p.drawRect(int(left_top.x()), int(left_top.y()), int(rect_width), int(rect_height))
527
528         if self.drawing() and not self.prev_point.isNull() and not self.out of pixmap(self.prev_point):
529             p.setPen(QColor(0, 0, 0))
530             p.drawLine(int(self.prev_point.x()), 0, int(self.prev_point.x()), int(self.pixmap.height()))
531             p.drawLine(0, int(self.prev_point.y()), int(self.pixmap.width()), int(self.prev_point.y()))
532
533         self.setAutoFillBackground(True)
534         if self.verified:
535             self.setPalette(QPalette())
```

LABELIMG ERROR CORRECTION

```
TypeError: setValue(self, int): argument 1 has unexpected type 'float'
• rokey-kim:~/Documents/ros2_ws$ labelImg
Traceback (most recent call last):
  File "/home/rokey-kim/.local/lib/python3.10/site-packages/labelImg/labelImg.py", line 965, in scroll_request
    bar.setValue(int(bar.value()) + bar.singleStep() * units)
TypeError: setValue(self, int): argument 1 has unexpected type 'float'
Traceback (most recent call last):
  File "/home/rokey-kim/.local/lib/python3.10/site-packages/labelImg/labelImg.py", line 965, in scroll_request
    bar.setValue(int(bar.value()) + bar.singleStep() * units)
TypeError: setValue(self, int): argument 1 has unexpected type 'float'
Traceback (most recent call last):
```

```
964         bar = self.scroll_bars[orientation]
965         bar.setValue(int(bar.value()) + int(bar.singleStep()) * int(units))
966
```

CODING HINTS

- Image Capture
- Data Labelling

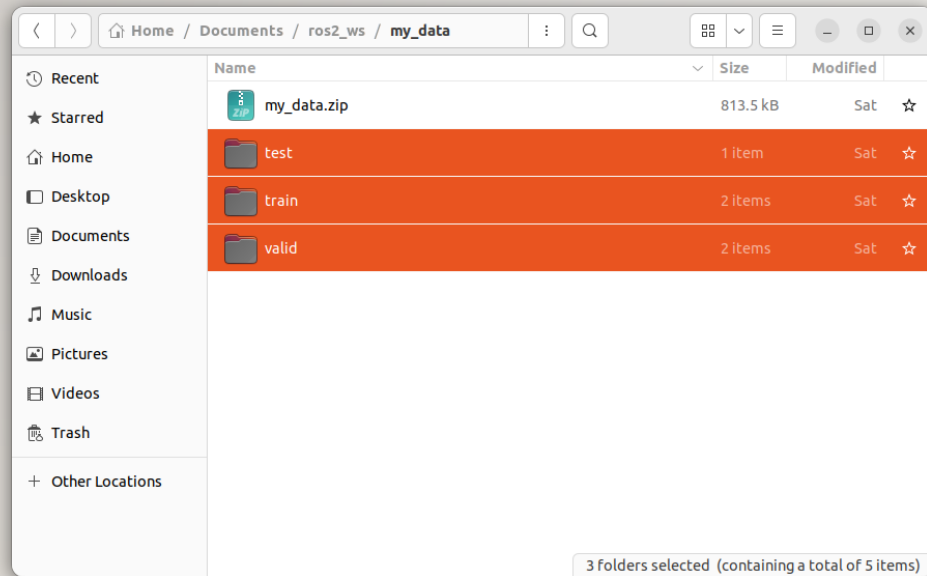
- Data Preprocessing

▼ day2

• [__init__.py](#)
• [2_1_a_capture_wc_image.py](#)
• [2_1_b_cont_capture_wc_image.py](#)
• [2_1_c_capture_wc_thread.py](#)
• [2_3_a_create_data_dirs.py](#)
• [2_3_b_move_image.py](#)
• [2_3_c_move_labels.py](#)



ZIP TRAIN DATA SET



PERFORM YOLO TRAINING & INFERENCE FOR DETECTION ALERT



CODING HINTS

- Image Capture
- Data Labelling
- Preprocessing

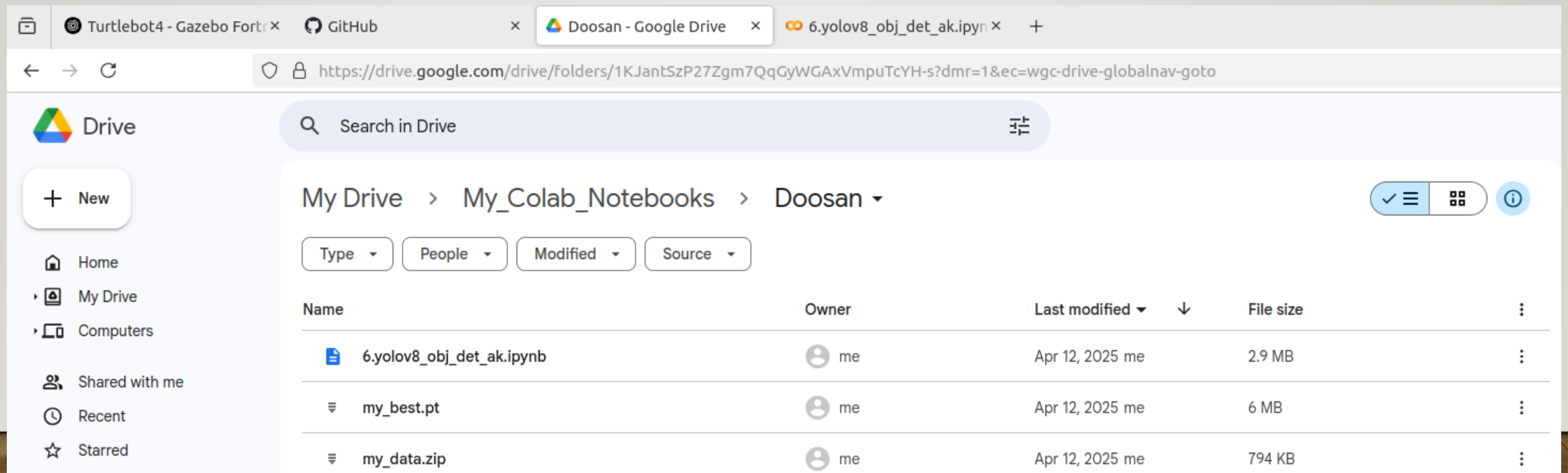
- Yolo8 Object Det



```
▼ day2
  + __init__.py
  + 2_1_a_capture_wc_image.py
  + 2_1_b_cont_capture_wc_image.py
  + 2_1_c_capture_wc_thread.py
  + 2_3_a_create_data_dirs.py
  + 2_3_b_move_image.py
  + 2_3_c_move_labels.py
  + 2_4_a_yolov8_obj_det_ak.ipynb
  + 2_4_b_gpu_test.py
  + 2_4_c_compare_yolo.py
  + 2_4_d_yolov8_obj_det_wc.py
```

USING GOOGLE COLLAB.TO CREATE CUSTOM MODEL

- Move the files to google drive
 - my_data.zip
 - yolov8.obj.det.ak.ipynb



The screenshot shows a web browser with multiple tabs. The active tab is 'Doosan - Google Drive'. The address bar shows the URL: <https://drive.google.com/drive/folders/1KJantSzP27Zgm7QqGyWGAXVmpuTcYH-s?dmr=1&ec=wgc-drive-globalnav-goto>. The Google Drive interface is displayed, showing a sidebar with navigation options: Home, My Drive, Computers, Shared with me, Recent, and Starred. The main area shows the 'My Drive' view, specifically the 'My_Colab_Notebooks' folder, and then the 'Doosan' sub-folder. A search bar is at the top. Below the folder path, there are filters for Type, People, Modified, and Source. A table lists the files in the folder:

Name	Owner	Last modified	File size
6.yolov8_obj_det_ak.ipynb	me	Apr 12, 2025 me	2.9 MB
my_best.pt	me	Apr 12, 2025 me	6 MB
my_data.zip	me	Apr 12, 2025 me	794 KB

USING GOOGLE COLLAB.TO CREATE CUSTOM MODEL

- Move the training script to google collab. and execute line by line

```
1 from google.colab import drive
2 drive.mount('/content/drive')

Mounted at /content/drive

1 # drive.flush_and_unmount()

yolov8 훈련하기 시작

dataset을 저장할 폴더를 만들고 그곳으로 들어갑니다

1 %mkdir dataset

Double-click (or enter) to edit

1 !cp /content/drive/MyDrive/My_Colab_Notebooks/Doosan/my_data.zip /content/

1 !unzip /content/my_data.zip -d dataset/

Archive: /content/my_data.zip
creating: dataset/test/
creating: dataset/test/images/
inflating: dataset/test/images/test_img_8.jpg
inflating: dataset/test/images/test_img_19.jpg
inflating: dataset/test/images/test_img_22.jpg
inflating: dataset/test/images/test_img_13.jpg
inflating: dataset/test/images/test_img_25.jpg
creating: dataset/train/
creating: dataset/train/images/
inflating: dataset/train/images/test_img_23.jpg

Connected to Python 3 Google Compute Engine backend (GPU)
```

SETTING UP YOLO

\$ pip install --upgrade pip

\$ nvidia-smi # Check if the GPU is available

If you have a GPU, install the CUDA version of PyTorch

\$ pip install torch torchvision torchaudio --extra-index-url <https://download.pytorch.org/whl/cu124>

\$ pip install ultralytics

\$ pip install "numpy<1.25.0"

REQUIRED PACKAGES SETUP

\$ pip list | grep opencv

If doesn't exist....

\$ pip3 install opencv-python

\$ pip3 install opencv-contrib-python

\$ pip list | grep ultra

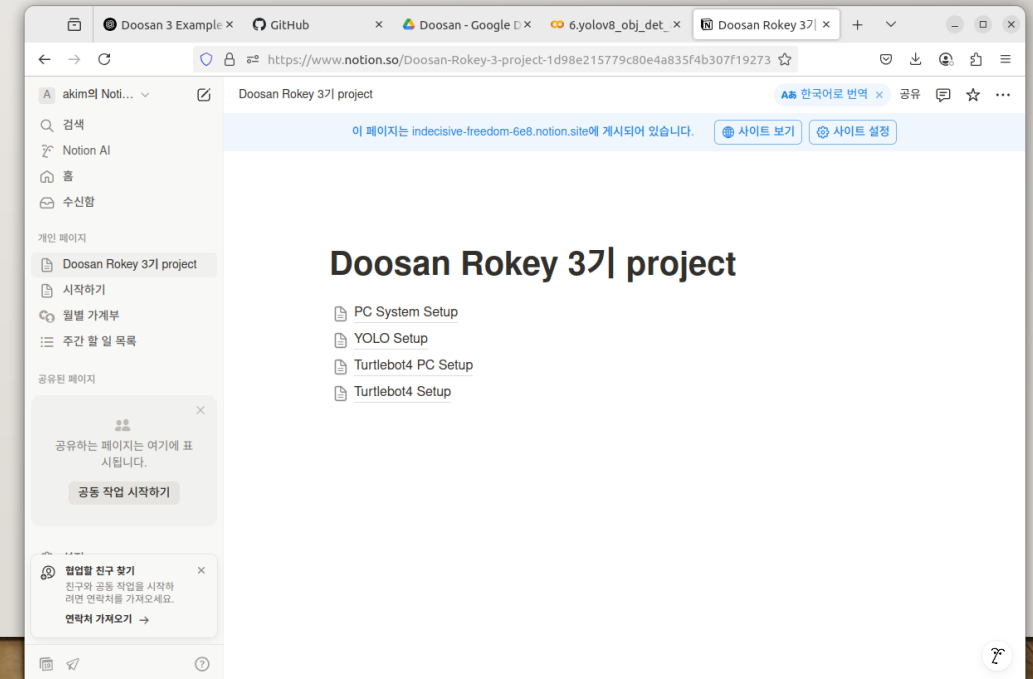
If doesn't exist....

\$ pip install ultralytics

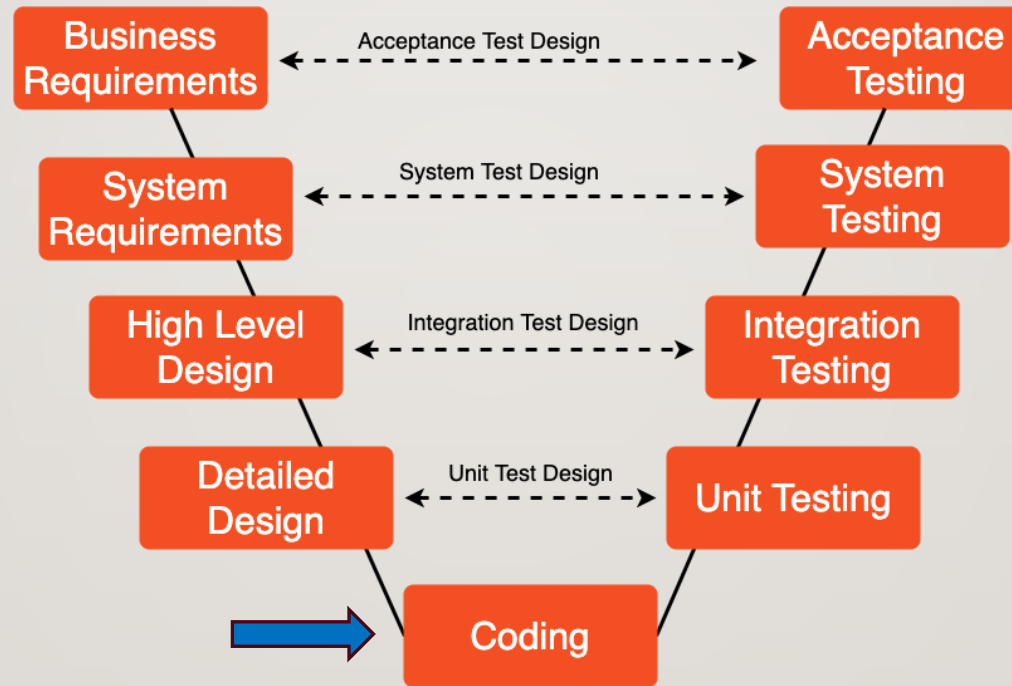


INSTALLING YOLO

- YOLO Setup
- <https://indecisive-freedom-6e8.notion.site/YOLO-Setup-1d98e215779c80f389eefbe0d86ee0ec>



SPRINT I - DETECTION ALERT



SDLC - V Model - notepub.io

HOW DID YOU/DO YOU NEED TO
DEFINE A DETECTION AREA?



CODING HINTS

- Image Capture
- Data Labelling
- Preprocessing
- Yolo8 Object Det
- Create your detection/alert condition



```
▼ day2
  🔗 __init__.py
  🔗 2_1_a_capture_wc_image.py
  🔗 2_1_b_cont_capture_wc_image.py
  🔗 2_1_c_capture_wc_thread.py
  🔗 2_3_a_create_data_dirs.py
  🔗 2_3_b_move_image.py
  🔗 2_3_c_move_labels.py
  📁 2_4_a_yolov8_obj_det_ak.ipynb
  🔗 2_4_b_gpu_test.py
  🔗 2_4_c_compare_yolo.py
  🔗 2_4_d_yolov8_obj_det_wc.py
  🔗 2_5_a_Draw_Box_wc.py
  🔗 2_5_b_Draw_Polygon_wc.py
  🔗 2_5_c_Security_Alert_wc.py
```

PORTING TO ROS

ROS2 DEVELOPMENT WORKSPACE

CREATE WORKSPACE

```
$ mkdir -p  
  ~/rokey3_<grp_letter><grp_num>_ws/src  
  • (i.e. mkdir -p ~/rokey3_A2_ws/src)
```

```
$ cd ~/rokey3_A2_ws
```

```
$ rosdep install --from-paths src --ignore-src  
-r -y
```

If not installed...

```
$ sudo rosdep init
```

```
$ rosdep update
```

*NOT CREATED UNTIL COLCON

```
workspace/      # Root of the workspace  
├─ src/         # Source code (ROS packages)  
├─ build/       # Build files (generated by colcon)  
├─ install/     # Installed packages and setup scripts  
└─ log/         # Build logs
```

```
$ colcon build
```

```
$ source install/setup.bash
```

ROS2 DEVELOPMENT WORKSPACE

```
$ cd ~/rokey3_A2_ws/src
```

```
$ ros2 pkg create --build-type  
ament_python <my_package>
```

```
my_package/  
├─ package.xml          # Package metadata and dependencies  
├─ setup.py             # Build instructions for Python packages  
├─ setup.cfg            # Optional, configures metadata for setuptools  
├─ launch/              # Launch files for starting nodes (optional)  
├─ config/              # Configuration files (optional)  
├─ resource/            # Empty file matching package name for ament index  
├─ my_package/          # Python package directory (contains code)  
│   └─ __init__.py      # Makes this directory a Python package  
│   └─ my_node.py       # Example Python node  
└─ msg/                 # Message definitions (optional)
```

ROS2 DEVELOPMENT WORKSPACE

Write your code below the `my_package/` directory under `my_package/ package` directory

```
my_package/
├─ package.xml          # Package metadata and dependencies
├─ setup.py             # Build instructions for Python packages
├─ setup.cfg            # Optional, configures metadata for setuptools
├─ launch/              # Launch files for starting nodes (optional)
├─ config/              # Configuration files (optional)
├─ resource/            # Empty file matching package name forament inc
├─ my_package/          # Python package directory (contains code)
│   └─ __init__.py      # Makes this directory a Python package
│       └─ my_node.py   # Example Python node
└─ msg/                 # Message definitions (optional)
```

ROS2 DEVELOPMENT WORKSPACE

```
$ cd ~/ rokey3_A2_ws
```

```
$ colcon build
```

```
workspace/      # Root of the workspace
├─ src/         # Source code (ROS packages)
├─ build/       # Build files (generated by colcon)
├─ install/     # Installed packages and setup scripts
└─ log/         # Build logs
```

```
$ echo "source ~/ rokey3_A2_ws/install/setup.bash" >> ~/.bashrc #check path
```

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


SETUP BASH

- `ROS_DOMAIN_ID = 0` in `.bashrc`
- `source ~/.bashrc`
- `ros2 daemon stop; ros2 daemon start`
- Make sure `discovery setup.bash` is not sourced!

CODING HINTS

create your team working folder and place
files and work from here

```
$ mkdir  
~/rokey3_<grp_letter><grp_num>_ws
```

(i.e. mkdir ~/rokey2_A2_ws)

ROS HINTS

```
▼ day2
  🔗 __init__.py
  🔗 2_1_a_capture_wc_image.py
  🔗 2_1_b_cont_capture_wc_image.py
  🔗 2_1_c_capture_wc_thread.py
  🔗 2_3_a_create_data_dirs.py
  🔗 2_3_b_move_image.py
  🔗 2_3_c_move_labels.py
  🔗 2_4_a_yolov8_obj_det_ak.ipynb
  🔗 2_4_b_gpu_test.py
  🔗 2_4_c_compare_yolo.py
  🔗 2_4_d_yolov8_obj_det_wc.py
  🔗 2_5_a_Draw_Box_wc.py
  🔗 2_5_b_Draw_Polygon_wc.py
  🔗 2_5_c_Security_Alert_wc.py
  🔗 2_6_a_image_publisher.py
  🔗 2_6_b_image_subscriber.py
  🔗 2_6_c_data_publisher.py
  🔗 2_6_d_data_subscriber.py
  🔗 2_6_e_yolo_publisher.py
  🔗 2_6_f_yolo_subscriber.py
  🔗 bus.ino
```

```
🔗 2_6_a_image_publisher.py
🔗 2_6_b_image_subscriber.py
🔗 2_6_c_data_publisher.py
🔗 2_6_d_data_subscriber.py
🔗 2_6_e_yolo_publisher.py
🔗 2_6_f_yolo_subscriber.py
🔗 bus.ino
```

ROS HINTS

- Edit setup.py under <package_name> directory add entry for each node

```
entry_points={ 'console_scripts':  
[ '<command_name> =  
<package_name>.<code_filename>:main', },
```

<command_name> is used when ros2 run is executed i.e. data_publisher

```
entry_points={  
    'console_scripts': [  
        'pub_image = day2.2_6_a_image_publisher:main',  
        'show_image = day2.2_6_b_image_subscriber:main',  
        'pub_data = day2.2_6_c_data_publisher:main',  
        'show_data = day2.2_6_d_data_subscriber:main',  
        'pub_yolo = day2.2_6_e_yolo_publisher:main',  
        'show_yolo = day2.2_6_f_yolo_subscriber:main',  
    ],  
}
```


ROS HINTS

\$ cd ~/rokey3_A2_ws

\$ sudo apt update

\$ source
~/rokey1_A2_ws/install/setup.bash

\$ sudo apt install terminator

\$ ros2 run <package_name>
<command_name>

```
1998  ros2 run day2 show_yolo
```

```
1999  ros2 run day2 pub_yolo
```

```
2000  1
```

ROS HINTS

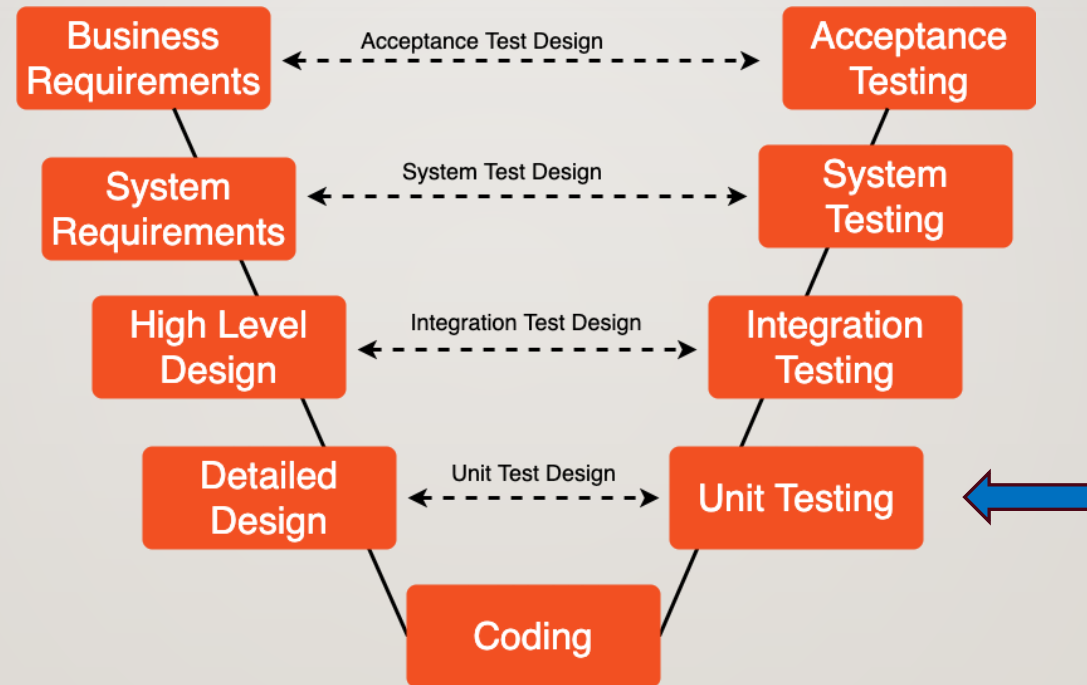
```
2_6_a_image_publisher.py
2_6_b_image_subscriber.py
2_6_c_data_publisher.py
2_6_d_data_subscriber.py
2_6_e_yolo_publisher.py
2_6_f_yolo_subscriber.py
bus.ino
```

```
$ ros2 run rqt_graph rqt_graph
$ ros2 node list
$ ros2 node info <node_name>
$ ros2 topic list
$ ros2 topic info <topic_name>
$ ros2 topic echo /chatter
$ ros2 interface list
$ ros2 interface show
  <package_name>/msg/<MessageName>
```

TEAM EXERCISE 5

Perform coding and testing of Detection Alert Module

SPRINT I - DETECTION ALERT



SDLC - V Model - notepub.io

EXPECTED OUTCOME

- Successful object detection
- ROS Nodes, and Topics created to send and display images and data

RESULTS & CODE REVIEW BY EACH TEAM

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

프로젝트 RULE NUMBER ONE!!!

Are we still having
FUN!

