

**CAPSTONE PROJECT REPORT**

**Report 4 – Software Design Document**

– Ho Chi Minh, October 2022 –

**Table of Contents**

[I. Record of Changes 3](#_heading=h.gjdgxs)

[II. Software Design Document 4](#_heading=h.30j0zll)

[1. System Design 4](#_heading=h.1fob9te)

[1.1 System Architecture 4](#_heading=h.3znysh7)

[1.2 Package Diagram 4](#_heading=h.2et92p0)

[2. Database Design 4](#_heading=h.tyjcwt)

[3. Detailed Design 5](#_heading=h.3dy6vkm)

[3.1 <Feature/Function Name1> 5](#_heading=h.1t3h5sf)

[3.2 <Feature/Function Name2> 6](#_heading=h.4d34og8)

# I. Record of Changes

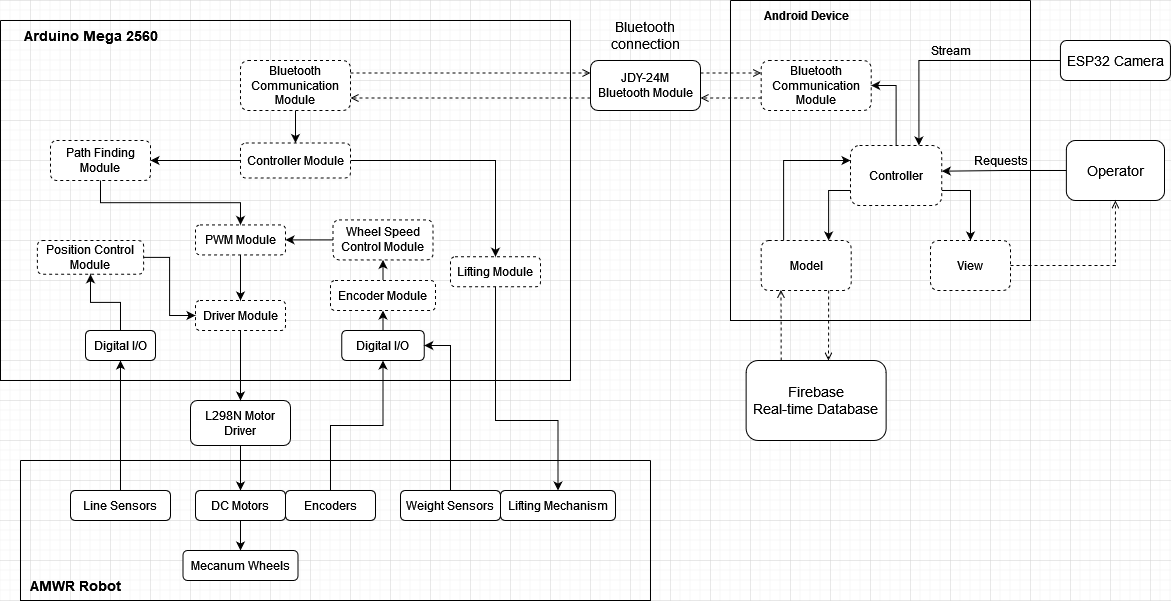
| **Date** | **A\* M, D** | **In charge** | **Change Description** |
| --- | --- | --- | --- |
| 19/10/2022 | A | Nguyễn Huỳnh Nhật Minh | Create initial draft |
|  |  |  |  |

\*A - Added M - Modified D - Deleted

# II. Software Design Document

## 1. System Design

### 1.1 System Architecture

*[The content of this section includes the overall diagram which includes the sub-systems, the external systems, and the relationship/connection among them. You need also provide the explanation for each of the diagram components (modules, sub-systems, external systems, etc.)]*

**AMWR System Architecture**

The system used two processing systems. The higher processor is an android device. It is responsible for processing instructions from operator inputs and image processing recorded from the ESP32 Camera. The lower-level processor is an Arduino Mega 2560, based on the ATmega 2560. It received commands from the android application and sent the specific commands for each motor through the DC motor driver. The Arduino is also responsible for reading the line sensors and controlling the individual DC motor to adjust the robot's position. The Arduino board communicates with the higher-level processor by bluetooth module JDY-24M and is a serial communication protocol.

The Arduino is also responsible for reading the encoders and controlling the individual DC motor speed. The speed of the robot is affected by the product’s weight detected by weight sensors. From the HIGH/LOW value detected by the Line Sensors, the Position Controller Module send the Driver Module in which direction the robot should move. The Driver Module receives direction and set the PWM (Pulse Width Modulation) value for each of the DC motors. The L298N Motor Driver act as an H-bridge, responsible for converting command from Arduino Mega to control the direction of the motors.

The Operator is responsible for input instructions. Inputs are processed by the Controller within the app and are sent through serial communication protocol for the robot to receive. Images of the loading area are recorded by an ESP32 Camera and processed by an android device to determine which spaces in the loading area are empty or occupied. It was sent to Controller to display real-time video. Instructions and information about the robot are logged for easy maintenance and check-up.

### 1.2 Package Diagram

*[Provide the package diagram for each sub-system. The content of this section includes overall package diagram(s) and the explanation for each package (or namespace). Please see the sample and description table format below]*



***Package Descriptions***

| **No** | **Package** | **Description** |
| --- | --- | --- |
| 01 | <Package name> | <Description of the package> |
| 02 |  |  |

## 2. Database Design

***Table Descriptions***

| **No** | **Table** | **Description** |
| --- | --- | --- |
| 01 | Real-time Operation | Store data of location of the robot and product on loading area map, robot info and current task being carry out by robot in real-time  - Primary keys: operation\_id int NOT NULL  - Foreign keys:   * robot\_num int NOT NULL * map\_num int NOT NULL * camera\_num int NOT NULL   - Attributes:   * current\_task varchar(50) NULL * current\_robot\_speed double NULL * current\_robot\_location varchar(50) NULL * current\_product\_location varchar(50) NULL * current\_pickup\_location varchar(50) NULL * current\_dropoff\_location varchar(50) NULL |
| 02 | ESP32 Camera | Store data of camera name, resolution and brightness setting and also its ip address  - Primary keys: cam\_num int NOT NULL  - Attributes:   * cam\_name varchar(50) NULL * cam\_resolution varchar(50) NULL * cam\_brightness int NULL * cam\_ip varchar(50) NULL * cam\_status boolean NOT NULL |
| 03 | Map | Store data of 2D map of the loading area in blob datatype  - Primary keys: map\_num int NOT NULL  - Attribute:   * map\_info blob NULL |
| 04 | Task | Store data of task that the robot can carry out  - Primary keys: task\_num int NOT NULL  - Attributes:   * task\_name nvarchar(50) NOT NULL |
| 05 | AMWR robot | Store data of robot info such as: name, status and bluetooth password  - Primary keys: robot\_ID int NOT NULL  - Attributes:   * robot\_name varchar(50) NOT NULL * robot\_status boolean NOT NULL * robot\_bluetooth\_password varchar(50) NULL |

## 3. Detailed Design

### 3.1 Show loading area map

### *[Provide the detailed design for the feature <Feature Name1>. It includes Class Diagram, Class Specifications, and Sequence Diagram(s);* ***For the features/functions with the same structure of class & sequence diagrams, you need to provide the diagrams once for one feature/function and refer to those diagrams from other features/functions****]*

#### 3.1.1 Class Diagram

*[This part presents the class diagram for the relevant feature]*



***3.1.2 <Sequence Diagram Name1>***

*[Provide the sequence diagram(s) for the feature, see the sample below]*



#### 3.1.3 <Sequence Diagram Name2>

#### 3.1.4 …

### 3.2 <Feature/Function Name2>

…