# Calliper Measurement System - Documentation

## 1. Introduction

This document provides a comprehensive overview of the Calliper Measurement System, including installation steps, package dependencies, CSS/UI refinements, backend implementations, and improvements made.

## 2. Tools and Packages Used

The following tools and packages were used in the implementation:

- Python 3.13 (Installed in C:/Users/Arbin/AppData/Local/Programs/Python/Python313/)

- Laravel 12.2.0

- Livewire (For dynamic UI updates)

- ImageMagick (For image processing)

- rembg (For background removal)

- JavaScript (For capturing and downloading UI images)

- Tailwind CSS (For responsive UI design)

## 3. Installation and Setup

Follow these steps to set up the system:

- Install Python and ensure it is added to the system PATH.

- Install Laravel and required PHP dependencies via Composer.

- Set up ImageMagick for image processing and configure PATH properly.

- Ensure rembg is installed using: `pip install rembg`

- Run Laravel migrations to set up the database.

- Configure environment variables in the `.env` file.

## 4. Backend Implementation

The backend is implemented using Laravel and processes the bead images using rembg and ImageMagick.

### Backend Flow Diagram

```mermaid  
graph TD;  
 A[User Uploads Image] -->|Sends Image| B[Backend Processing];  
 B -->|Remove Background| C[rembg Library];  
 C -->|Save Processed Image| D[Storage Folder];  
 D -->|Return Image URL| E[Frontend Display];  
```

## 5. UI Enhancements

The UI has been refined for better usability with the following improvements:

- Responsive layout using Tailwind CSS.

- Improved positioning of elements for better readability.

- Implemented proper image capture and download functionality.

- Ensured high-quality image rendering when saving from UI.

### UI Flow Diagram

```mermaid  
graph TD;  
 A[User Inputs Size & Image] --> B[Upload & Process];  
 B -->|Store Image| C[Storage Folder];  
 C -->|Display Processed Image| D[Frontend UI];  
 D -->|User Downloads Image| E[Save as PNG];  
```

## 6. JavaScript Image Capture Implementation

The following JavaScript snippet was used to capture and download the Calliper measurement UI:

```javascript  
function downloadImage() {  
 html2canvas(document.querySelector('.calliper-container')).then(canvas => {  
 let link = document.createElement('a');  
 link.download = 'calliper\_measurement.png';  
 link.href = canvas.toDataURL();  
 link.click();  
 });  
}  
```