Project2_Milestone2 Partially Functional Prototype:

Check milestone2 (functional, might have bugs)

Journal documentation & Roadblocks:

I chose to build the app with steps counts, though I thought others did but I never try it by myself. Right now, I have the basic layout of the whole app and working on add count method and step detector and listener successfully(which is the main point). Also, I want to make the history persistent and try to make a clearly count table to show the record data. Still need to add more flow and change the details of whole app. Also, I'm thinking to change the layout into the bottom navigation, need to listen suggestions from classmates which way they think is better.

Flow and Interactivity:

First it will show the homepage which contains the a counting graph of steps and steps number that you have today. Also, there are two buttons which are "Make Exercise plan" and "Check History." When you hit "Make Exercise plan," you will be lead to another page to set up your Daily exercise goal, reminder and set reminder time kind of things. If you save the change it will lead you back to the homepage. Then if you click on the "Check History" button, you will be lead to your steps history in a list form. However I hope to make it into a graph which will be more clearly to show your exercise progress.

Logic and Algorithm:

Motion Sensor:

https://developer.android.com/guide/topics/sensors/sensors_motion

The step detector sensor

https://hub.packtpub.com/step-detector-and-step-counters-sensors/

```
public class PedometerActivity extends Activity implements SensorEvent

  private SensorManager mSensorManager;
  private Sensor mSensor;
  private boolean isSensorPresent = false;
  private TextView mStepsSinceReboot;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_pedometer);

        mStepsSinceReboot =
        (TextView)findViewById(R.id.stepssincereboot);

        mSensorManager = (SensorManager)
        this.getSystemService(Context.SENSOR_SERVICE);
    if(mSensorManager.getDefaultSensor(Sensor.TYPE_STEP_COUNTER)
    != null)
    {
      mSensor =
      mSensorManager.getDefaultSensor(Sensor.TYPE_STEP_COUNTER);
      isSensorPresent = true;
    }
    else
    {
      isSensorPresent = false;
    }

    }

    @Override
  protected void onResume() {
    super.onResume();
    if(isSensorPresent)
```

```
    @Override
  protected void onResume() {
    super.onResume();
    if(isSensorPresent)
    {
      mSensorManager.registerListener(this, mSensor,
      SensorManager.SENSOR_DELAY_NORMAL);
    }
  }

  @Override
  protected void onPause() {
    super.onPause();
    if(isSensorPresent)
    {
      mSensorManager.unregisterListener(this);
    }
  }

@Override
  public void onSensorChanged(SensorEvent event) {
    mStepsSinceReboot.setText(String.valueOf(event.values[0]));

  }
```