The University Of Sheffield.

COM4510/6510
Software Development for Mobile Devices

Lecture 10: Releasing Apps over large scale

Dr Po Yang
The University of Sheffield
po.yang@sheffield.ac.uk

Lecture Overview

- Part 1:
  - Releasing an app over large scale

- Part 2:
  - Examples of apps we have developed

- Lab tutorial :
  - Preparing assignments

Releasing an app over large scale

Challenges: Building

- **Professional <u>engineering</u> infrastructure**
  - **Development tools and servers**
    - **You do not build a house by just pulling up walls!**
- **Professional software <u>engineers</u>**
  - **Using professional software <u>engineering</u> tools**

image from the http://www.birminghamupdate

Seeing with the users' eyes

- You cannot have an app on the market and just wait for reviews
- You must
  - be able to **see the problems** before they become issues for users
  - see the **stack trace of errors** that happen on phones (Java or Kotlin)
  - see **ANRs** (Application not Responding)
  - The **patterns of installation/uninstallation**
  - the time spent on your app
  - the level of involvement of users

The University Of Sheffield.

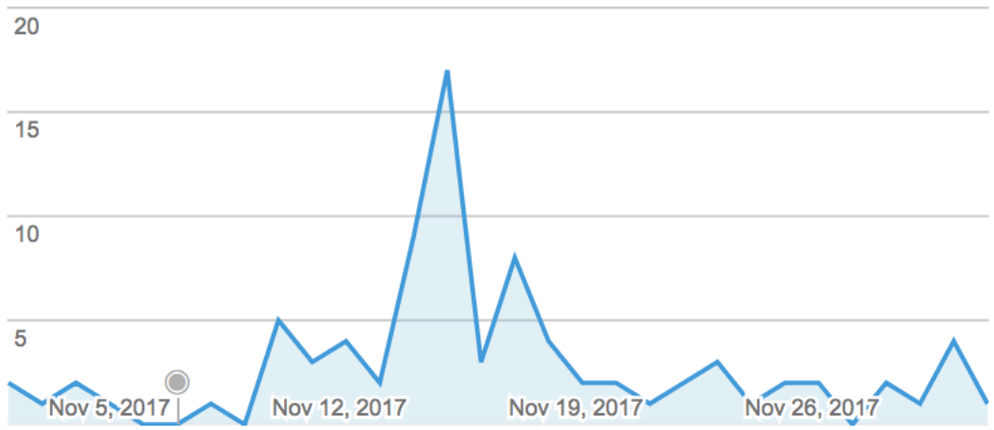
Google Play Dashboard

7 DAYS | 30 DAYS | 1 YEAR | LIFETIME

Installs by user VIEW DETAILS

85
Last 30 days

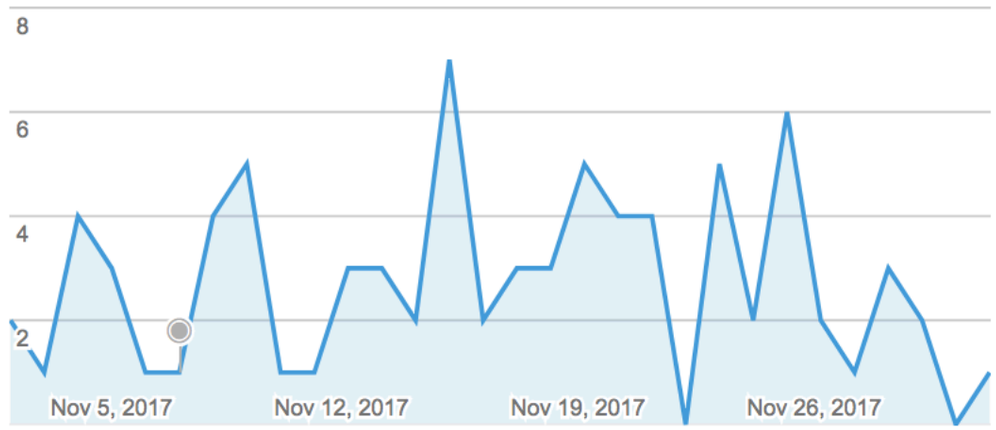
-20%
vs previous 30 days



Uninstalls by user VIEW DETAILS

81
Last 30 days

+59%
vs previous 30 days



Installs by user VIEW DETAILS

Top countries

Italy		76.0	+117.1%
United Kingdom		6.00	-68.4%
Hungary		1.00	
Indonesia		1.00	0.0%
Turkey		1.00	

The University Of Sheffield.

Installs On Active Devices by **Android Version**. Learn more

Compare to: 2 Oct 2017 – 31 Oct 2017

Final installs on active devices

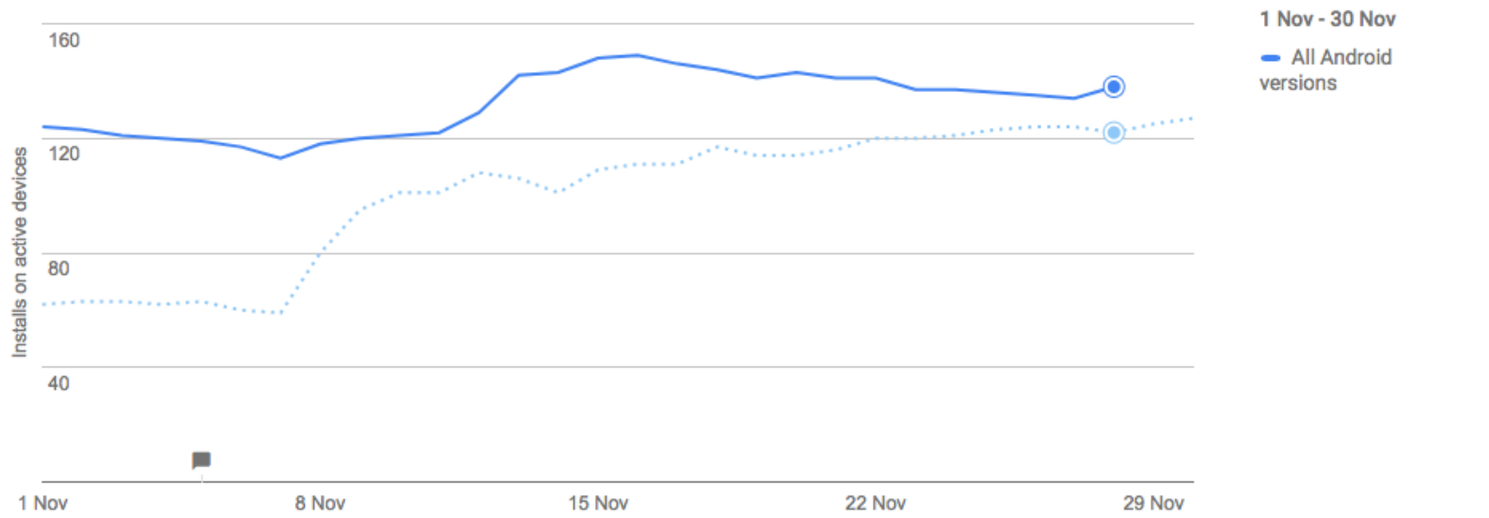
138

+8.7% vs previous 30 days

Download bulk reports

Installs on active devices

Daily intervals | Display date comparison



Search Android Version

Displayed on table

28 Nov 2017

Installs on active devices

Android Version	Total	% of total	+/- vs 29 Oct
All Android versions	138	100.0%	+13.1%
Android 7.0	51	37.0%	+4.1%
Android 6.0	43	31.2%	+7.5%
Android 5.1	26	18.8%	+85.7%
Android 5.0	9	6.5%	-10.0%
Android 8.0	4	2.9%	+33.3%
Android 7.1	3	2.2%	-25.0%
Android 4.4	2	1.4%	0.0%

The University Of Sheffield.

Real-time crashes

Last 7 days | All Android versions | Current production (...

5 crash clusters

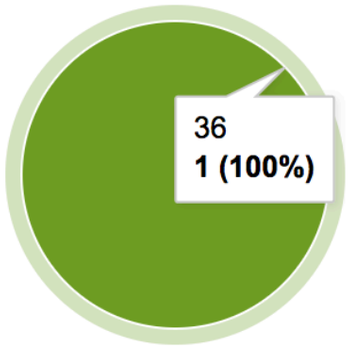
Show hidden

Cluster	Trends	Reports	Impacted users	Last reported
java.lang.NullPointerException in uk.ac.shef.oak.ActivityRecognition.WeSenseIt.Core.locationS...		200	2	27 Nov 20:56
New in version 36 java.lang.NullPointerException in oak.shef.ac.uk.abstractcityactivity.menu.MenuActivity.ifShow...		3	2	Yesterday, 22:22
New in version 36 java.lang.IllegalArgumentException in uk.ac.shef.oak.ActivityRecognition.WeSenseIt.Core.SensorSer...		2	1	26 Nov 05:29
java.lang.NullPointerException in oak.shef.ac.uk.abstractcityactivity.MainActivity.onArticleSelec...		1	1	24 Nov 11:52
New in version 36 java.lang.NullPointerException in <OR> uk.ac.shef.oak.ActivityRecognition.WeSenseIt.Core.data...		1	1	23 Nov 22:20

Page **1** of **1**

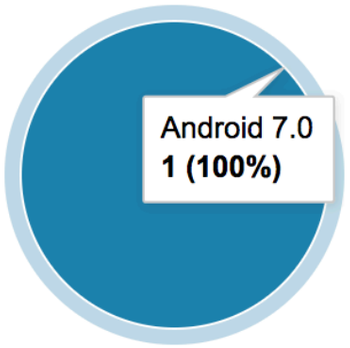
The University Of Sheffield.

By app version



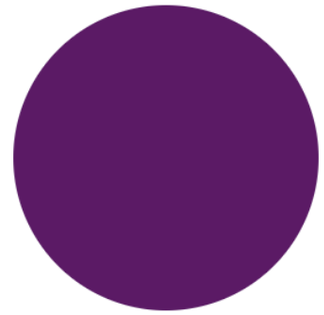
36	1	100.0%

By Android version



Android 7.0	1	100.0%

By device



P9 lite (HWVNS-H)	1	100.0%

23 Nov 22:20 on app version 36

Report **1** of **1**

Huawei P9 lite (HWVNS-H), 3072MB RAM, Android 7.0

java.lang.RuntimeException:

```
1.     at android.os.AsyncTask$3.done (AsyncTask.java:330)
2.     at java.util.concurrent.FutureTask.finishCompletion (FutureTask.java:354)
3.     at java.util.concurrent.FutureTask.setException (FutureTask.java:223)
4.     at java.util.concurrent.FutureTask.run (FutureTask.java:242)
5.     at android.os.AsyncTask$SerialExecutor$1.run (AsyncTask.java:255)
6.     at java.util.concurrent.ThreadPoolExecutor.runWorker (ThreadPoolExecutor.java:1133)
7.     at java.util.concurrent.ThreadPoolExecutor$Worker.run (ThreadPoolExecutor.java:607)
8.     at java.lang.Thread.run (Thread.java:776)
```

Caused by: **java.lang.NullPointerException:** at uk.ac.shef.oak.ActivityRecognition.WeSenseIt.Core.database.Database.insertSummary(Database.java)

```
1.     at <OR> uk.ac.shef.oak.ActivityRecognition.WeSenseIt.Core.database.Database.insertSummary (Database.java)
2.     at oak.shef.ac.uk.cityactivity.engineAPI.EngineAPI.computeAndUpdateCurrentProgress (EngineAPI.java)
3.     at oak.shef.ac.uk.cityactivity.engineAPI.EngineAPI.recomputeAndUpdateCurrentProgress (EngineAPI.java)
4.     at oak.shef.ac.uk.cityactivity.alarmreceivers.UploadDataTask.populateLastDayUnsentRecords (UploadDataTask.java)
```

The University Of Sheffield.

Fabric

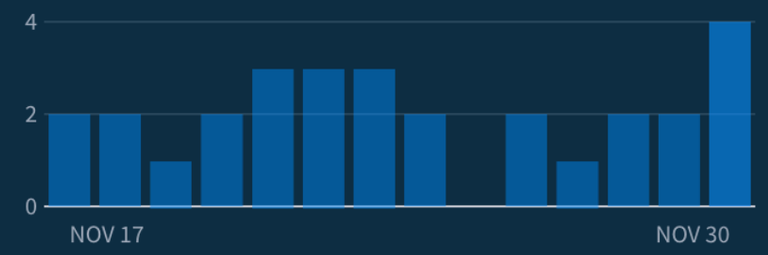
13 +44.4%

Daily Active Users

Retention

4

Daily New Users



Growth

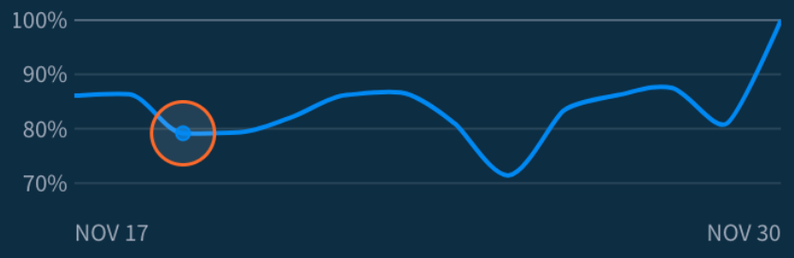
Set your KPI

Track key metrics for your app's performance.

Set your KPI

100.0%

Crash-Free Users for All Builds

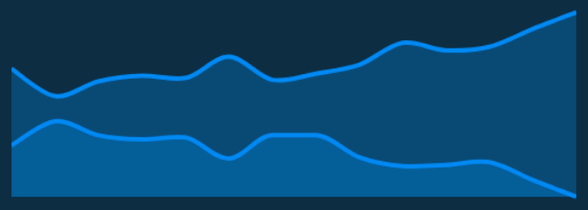


Build	Today
All Builds	100.0%
2.3.12 (36)	100.0%
2.3.11 (34)	—

Stability by Crashlytics

100.0%

Adoption of 2.3.12 (36)

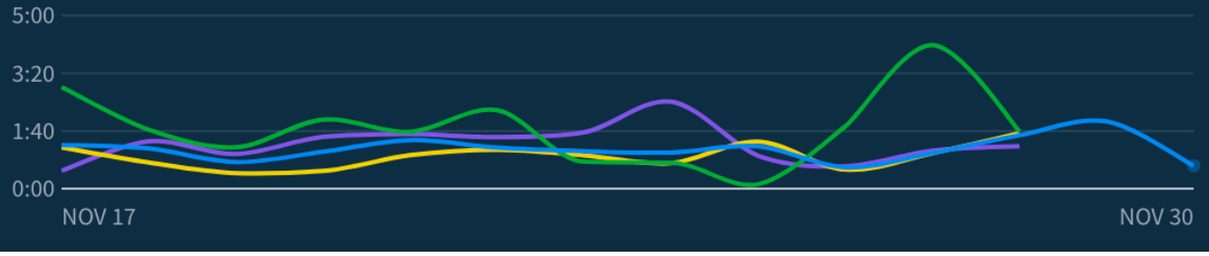


Build	Adoption	Stability
2.3.12 (36)	100.0%	100.0%
2.3.11 (34)	0.0%	100.0%

Latest Release

0:40 sec -48.1%

Median Total Time Spent in App Per User

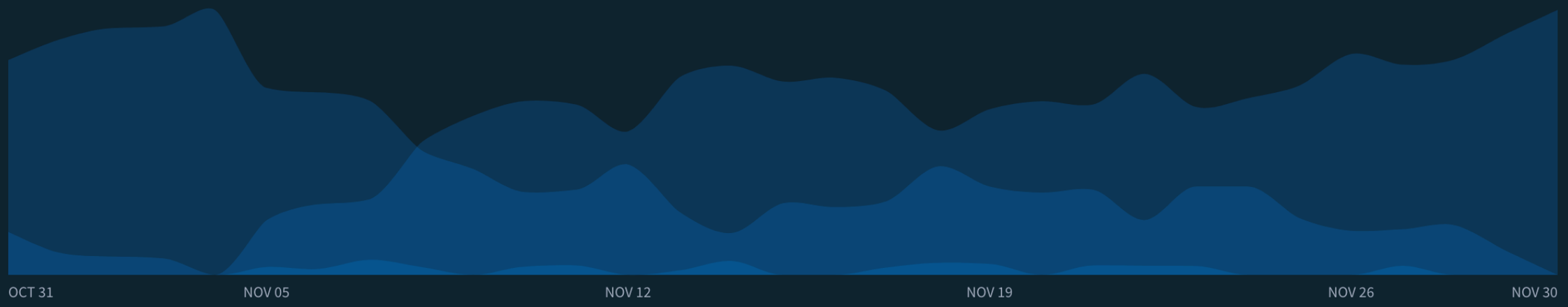


The University Of Sheffield.

Latest Release

Release 2.3.12 (36) has been **Successful**

With **100.0% crash-free users** and **100.0% of daily active users**



Top Builds

2.3.12 (36)

Successful Release

See more details

Adoption

100.0% of Total DAU

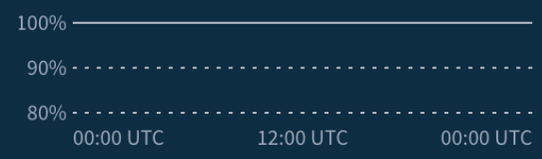
13 DAU on Build

0:18 sec Median Session Length

Stability

100.0% Crash-free Users

0 Crashes



Investigate this release's top issues in Crashlytics.

The University Of Sheffield.

Nov 27 2017 20:20 (UTC)

Version 2.3.12 (36)

All Versions

Device

Galaxy S6 Edge
DEVICE

Portrait
ORIENTATION

On
PROXIMITY

Operating System

7.0
OS VERSION

Portrait
UI ORIENTATION

No
ROOTED

Device Statistics

12%
Ram Free

61%
Disk Free

Keys Logs

No keys found. Learn more!

Stacktrace **Download .txt**

Fatal Exception: java.lang.NullPointerException
Attempt to invoke virtual method 'void uk.ac.shef.oak.ActivityRecognition.WeSenseIt.Core.locationService.MyLocationService.insertLocationWithExistingLocation(android.location.Location)' on a null object reference

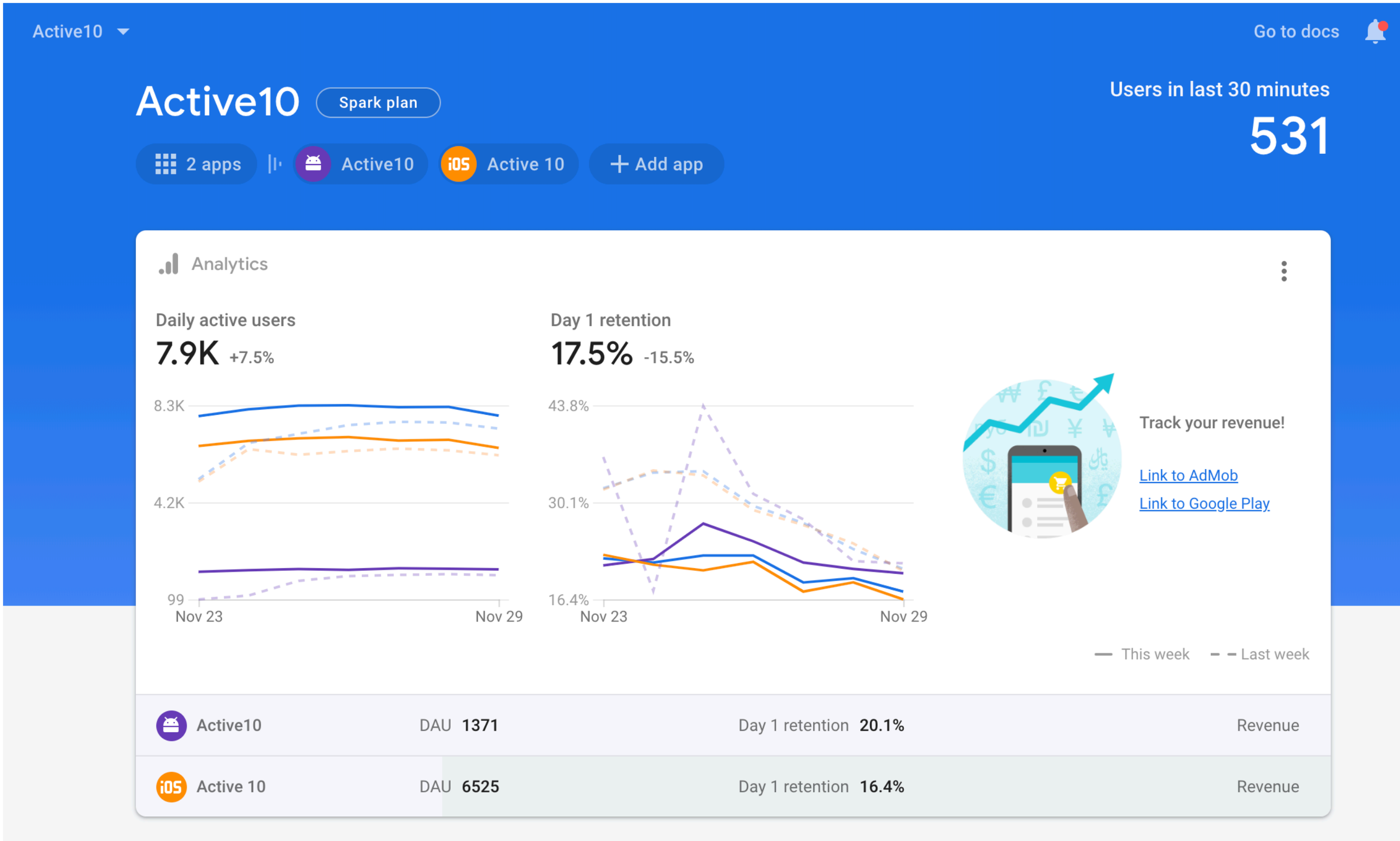
</> Raw Text

```
Fatal Exception: java.lang.NullPointerException: Attempt to invoke virtual method 'void uk.ac.shef.oak.ActivityRecognition.WeSenseIt.C
       at uk.ac.shef.oak.ActivityRecognition.WeSenseIt.Core.locationService.StandardAndroidLocationService$LocationIntent.onHandleInte
       at android.app.IntentService$ServiceHandler.handleMessage(IntentService.java:67)
       at android.os.Handler.dispatchMessage(Handler.java:102)
       at android.os.Looper.loop(Looper.java:154)
       at android.os.HandlerThread.run(HandlerThread.java:61)
```

The University Of Sheffield.

Firebase

Replacing Crashlytics



The University Of Sheffield.

Active10

Go to docs

Dashboard

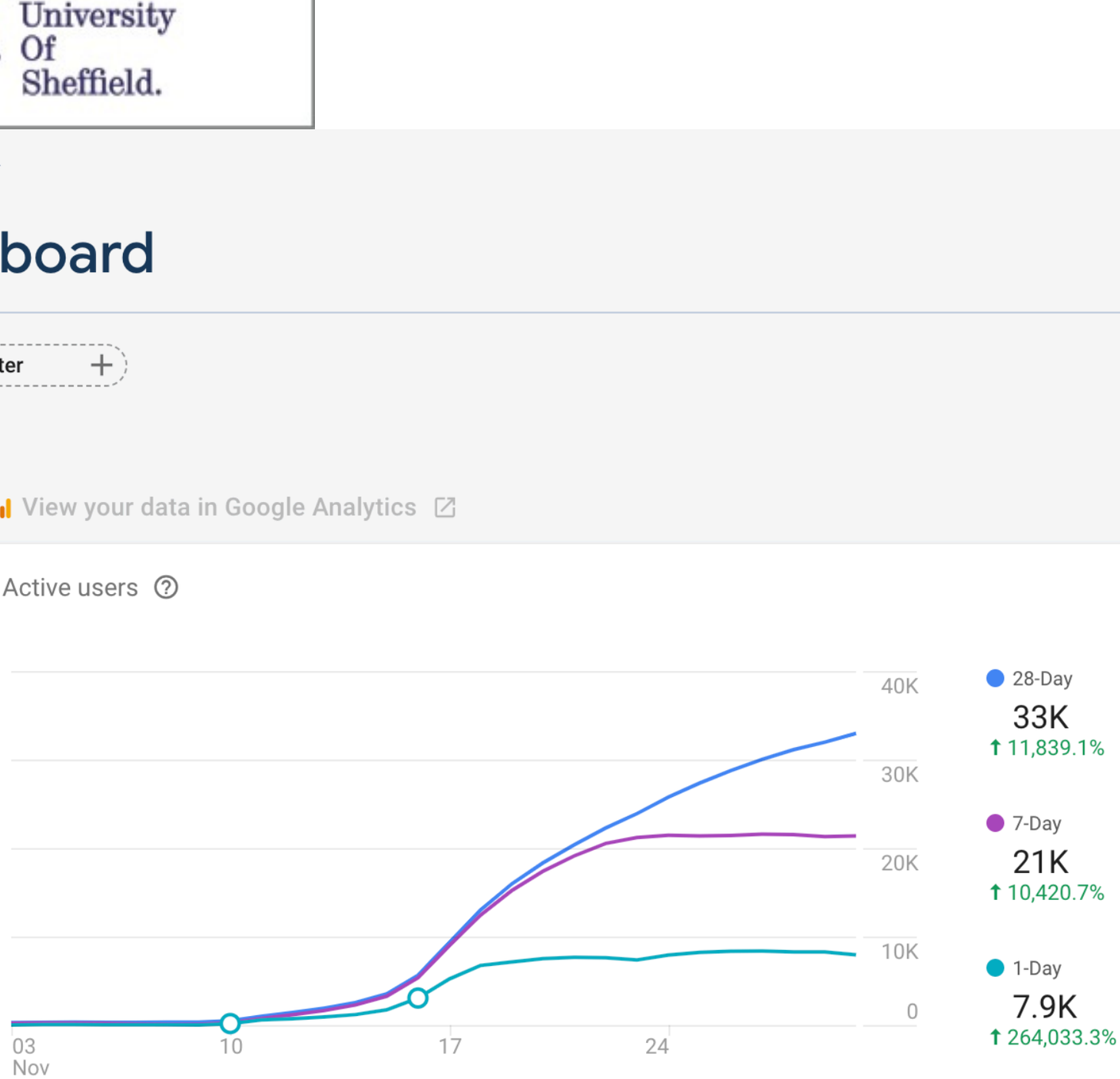
Add Filter

Last 28 days

Compared to Oct 6, 2019 - Nov 2, 2019

View your data in Google Analytics

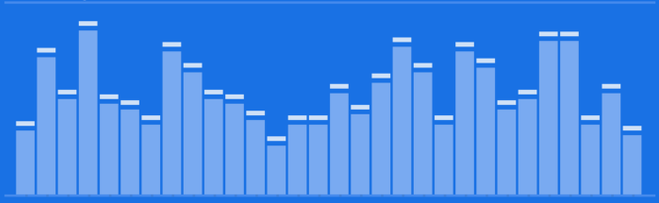
Active users



Users in last 30 minutes

536

Users per minute



Top conversion events	Count
first_open	27

StreamView

How often are your users converting?

Where are your users engaged?

Daily user engagement

The University Of Sheffield.

Active10 ▾ Go to docs

Dashboard

Add Filter +

Last 28 days ▾

Compared to Oct 6, 2019 - Nov 2, 2019

How well do you retain users?

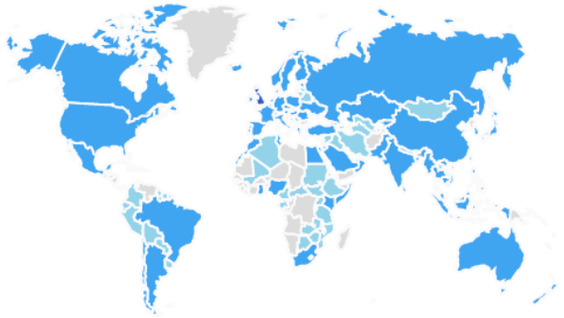
Retention cohorts ⓘ 6 weeks ending Nov 30 ⓘ

	Week 0	Week 1	Week 2	Week 3	Week 4	Week 5
All Users	100.0%	52.7%	50.0%	4.2%	2.8%	10.7%
Oct 20 - Oct 26						
Oct 27 - Nov 2						
Nov 3 - Nov 9						
Nov 10 - Nov 16						
Nov 17 - Nov 23						
Nov 24 - Nov 30						

View new user retention →

What is your audience like?

Location ⓘ Devices ⓘ Demographics ⓘ Interests ⓘ



Country/Region	Sessions	% Total
United Kingdom	299K	89.2%
United States	5.9K	1.7%
Australia	3.2K	1.0%

View "All Users" audience →

What is your platform breakdown?

Platforms

The University Of Sheffield.

Active10

Go to docs

Audiences

All Users

All users

Add Filter

Last 28 days

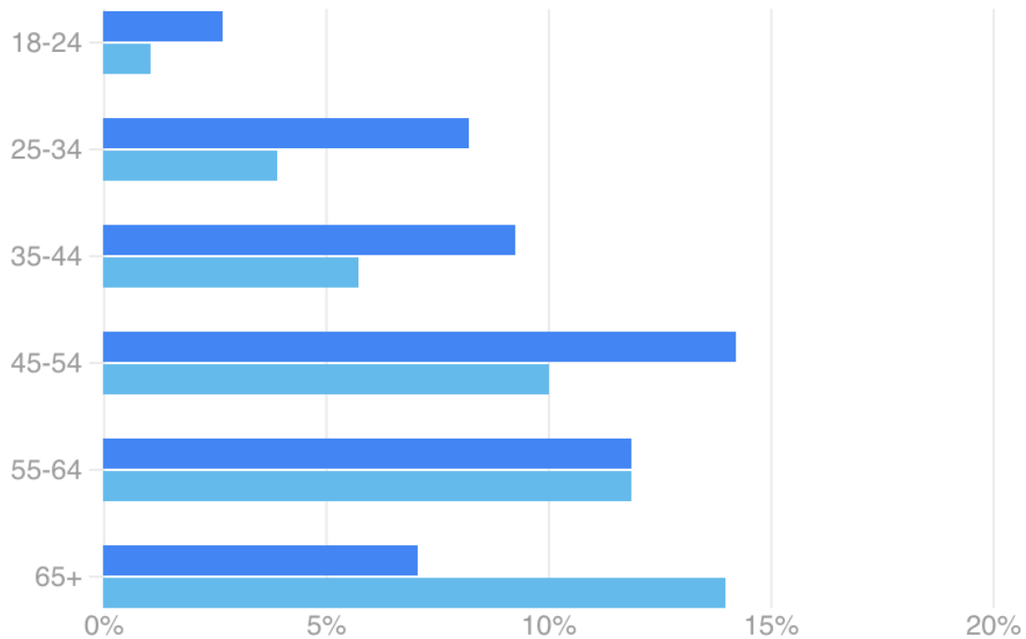
Compared to Oct 6, 2019 - Nov 2, 2019

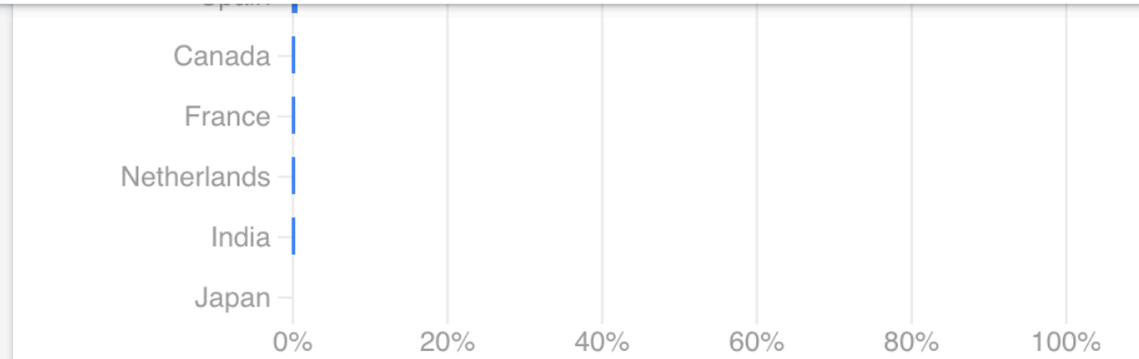
Gender

Female
53.6%
-

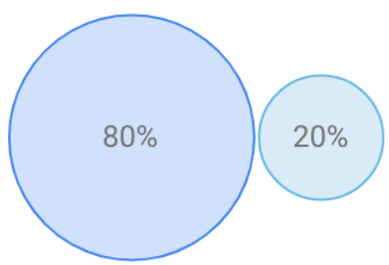
Male
46.4%
↑ 2,967.1%

Age





Platforms



Platform	% Total
iOS	80%
Android	20%

The University Of Sheffield.

Active10

Go to docs

Crashlytics

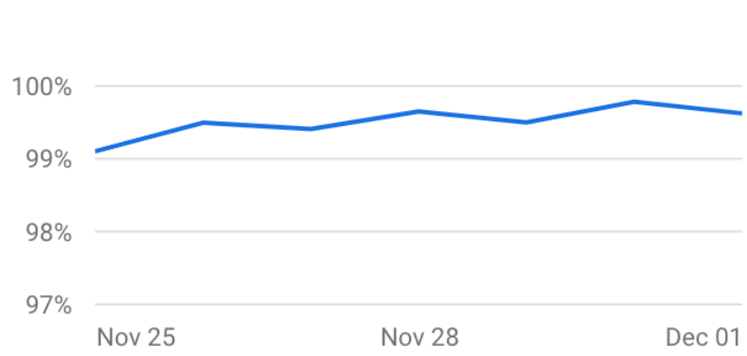
Active10

Filter | Event type = "Crashes"

Last 7 days
25 Nov 2019 – 1 Dec 2019

Crash-free statistics

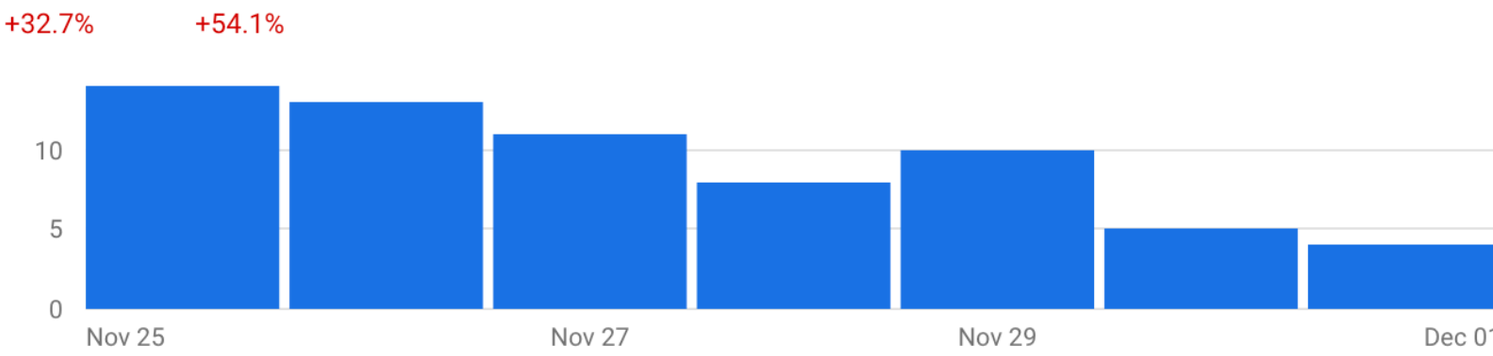
Crash-free users
98.99%



Event trends

Latest Release

Crashes	Users affected
65	57
+32.7%	+54.1%



View and analyse your raw crash data in BigQuery Link BigQuery Learn more

Issues

Search by user

Filter issues | Issue state = "Open"

Search issue title or subtitle

Releasing an app over large scale

Architecture Coverage

- **Iphone vs Android**
  - both must be covered
  - both required different programmers
  - no intersection of code
    - in MoveMore we saved on a common interface based on WebViews and we are still regretting it
- Android:
  - different vendors, quirk Android implementations
- iOS
  - Older phones not built for tracking
- How far back do you go?

The University Of Sheffield.

Risks

Risk	Impact	Likelihood	Impact	Mitigating Measures	Residual risk
Phone is held in places where it is subject to multiple forces (e.g. large bags, hands, etc.)	Sensors receive contradicting signals. This is a major source of potential error.	H	H	Clear instructions must be given to the user so that the phone is kept in contact with body	L
Phone is used for intensive activities e.g. gaming using on accelerometer-based interaction	Sensors receive contradicting signals - e.g. additional activities recognised	L – only some specific types of games can cause issue	H	- Strategies are implemented to counter spurious activities - Clear instructions should be given to the user - Enabling explicit user activity recognition interruption from interface (?)	L

Other activities

Some activities may interfere with activity recognition. In particular calling and texting can in some users heavily influence the way they walk	Sensors receive contradicting signals – the movement of a person walking while calling assumes very often all the characteristics of cycling.	M	H for people who make very frequent calls while walking	Strategies can be implemented to counter this issue but they are expensive battery wise so they will not be implemented initially. Users should be made aware of this potential issue.	M

The University Of Sheffield.

AVERAGE TIME SPENT PER PERSON BY PLATFORM IN DECEMBER 2013

	U.S.	U.K.	Italy
Monthly TV time spent	185 hours	129 hours, 54 minutes	143 hours, 20 minutes
Monthly online time spent	26 hours, 58 minutes	29 hours, 14 minutes	18 hours, 7 minutes
Monthly mobile time spent	34 hours, 21 minutes	41 hours, 42 minutes	37 hours, 12 minutes

SHARE OF TIME SPENT USING SMARTPHONE APPS BY CATEGORY

	U.S.	U.K.	Japan
Communications	12%	9%	16%
Shopping/Commerce	2%	5%	6%
News/Info	2%	5%	5%
Productivity/Function	11%	7%	6%
Entertainment	6%	15%	9%
Games	9%	18%	16%
Social	28%	29%	24%
Others	29%	12%	17%

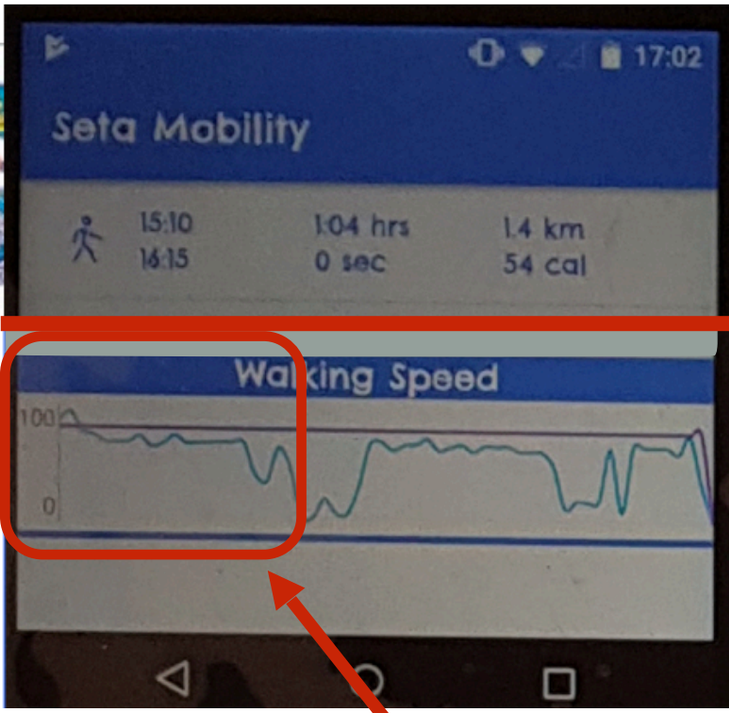
Source: Nielsen

Source: [Nielsen]

The University Of Sheffield.

Sensor Quality

Risk	Impact	Likelih	Impact	Mitigating Measures	Residual risk
Quality of sensors is low	Sensors return inaccurate output readings (e.g. miss to report movement or over-recognise movement)	L /M	H	Algorithm has built in strategies to cope with these situations	L –imprecision is still possible
Some relevant sensors are missing (e.g. internal step counter)	Activity recognition may be less accurate	L	M	Algorithm has built in strategies using multiple sensors	L - some functionality (e.g. step counting) will be unavailable if appropriate sensors are not present or non functional at a reasonable quality level – unless the raw accelerometer is used – a less reliable and more power hungry

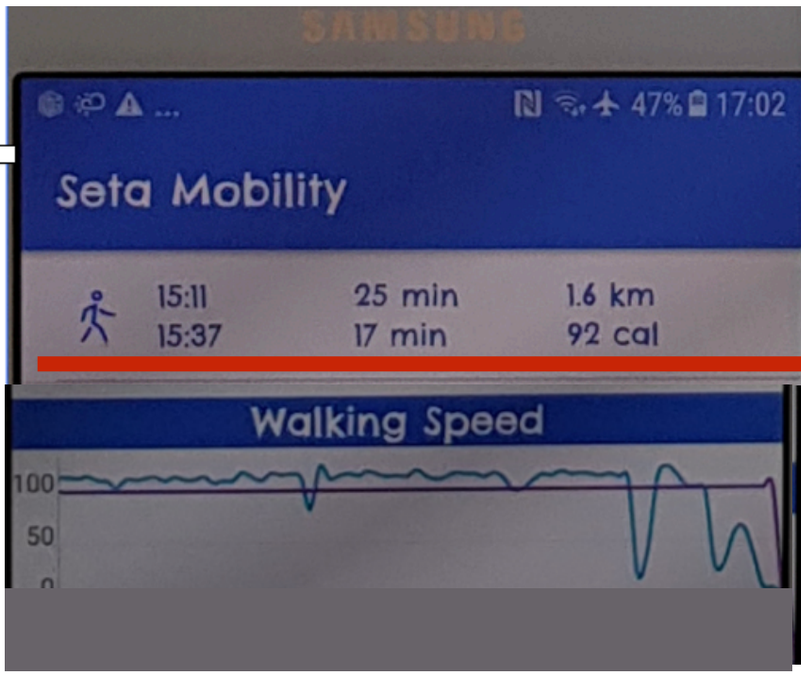


Moto G1 1/25

Samsung J5 17/20

Moto G 5 17/21

Samsung A5 17/25



LG L4 17/24

Huawei P9 16/21

The University Of Sheffield.

Power Management

Risk	Platform	Impact	Likelihood	Mitigating Measures	Residual Risk
Aggressive power and memory saving strategies kill the background process	A	H	H	Algorithm has built in strategies to aggressively restart the process if killed by operating system	L strategies are effective but not full proof as out of control of the app
Aggressive power saving strategies limit data availability to the phone	A	Sensors do not return output	L	Algorithm has built in strategies to cope with these situations by e.g. - fusing sensors of different types (e.g. wake up sensors Vs non wake up sensors)	L – strategies are effective but not full proof as outside the control of the app Some functions (e.g. step counting) will be unavailable –

Other Risks

					be high
Non standardised operating system level functions and strategies	Andr oid	It may difficult to control all situation s – some	H	Documentation is available for most vendors but the experience with Android 6 in 2016 has been rather painful.	M – the general opinion is that the strategies for Android have become stable and are now more and more followed by –

Risk	Platf orm	Impact	Likeli hood	Imp act	Mitigating Measures	Residua l Risk
No availability of health kit and associated low power sensors	iOS pre-5s	High battery usage	H	H	Algorithm tries to minimise battery usage but it is the architecture of the phone that is not designed for the task. These phones users should be discouraged (or prevented) from using the app	H
Slow power hungry processors	Early versio ns of A and old A phone s	High battery usage	H for those phones	H	Algorithm tries to minimise battery usage but it is the architecture of the phone that is not designed for the task. These phones users should be discouraged (or prevented) from using the app	H

The University Of Sheffield.

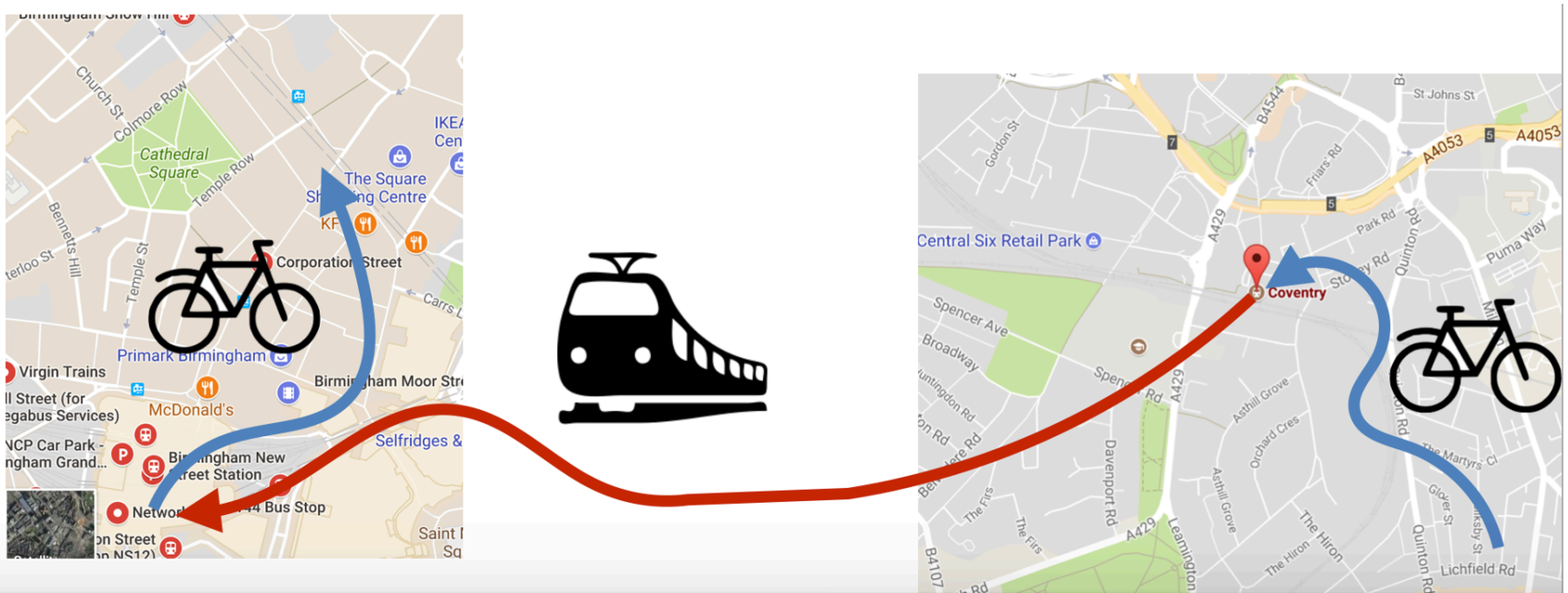
Examples of apps we have developed

Overviews

- All apps concern sensing human mobility (2017-2019)
  - **Active 10** - on behalf of Public Health England
    - over 870,000 downloads for Android+iOS
    - working on 10,000 device types and 6 versions of android (4.4+)
    - Just dismissed by PHE
  - **Seta** - on Behalf of Birmingham City Council (2017-current)
    - 8,000 users expected to double in the next couple of years
  - **Seta4Schools** (2018-2019)
    - release on Dec 1st 2018
    - 1,000 users in Santander, Spain in January 2019
  - **MoveMore Sheffield** (2016-2017)
    - about 6,000 users
  - **Aeqora** (beta 2019 - 3 Hospitals in UK, Germany and Israel in 2020)
- Future:
  - Skateboarding England (2020)
  - NHS - Tracking home mobility in patients (beta 2019 - with patients 2020)

Objectives

- Tracking multimodal mobility
  - walking, running cycling, use of vehicles
  - to model behavioural aspects of modern living
    - health and wellbeing, mobility and transport
  - to support



Approach

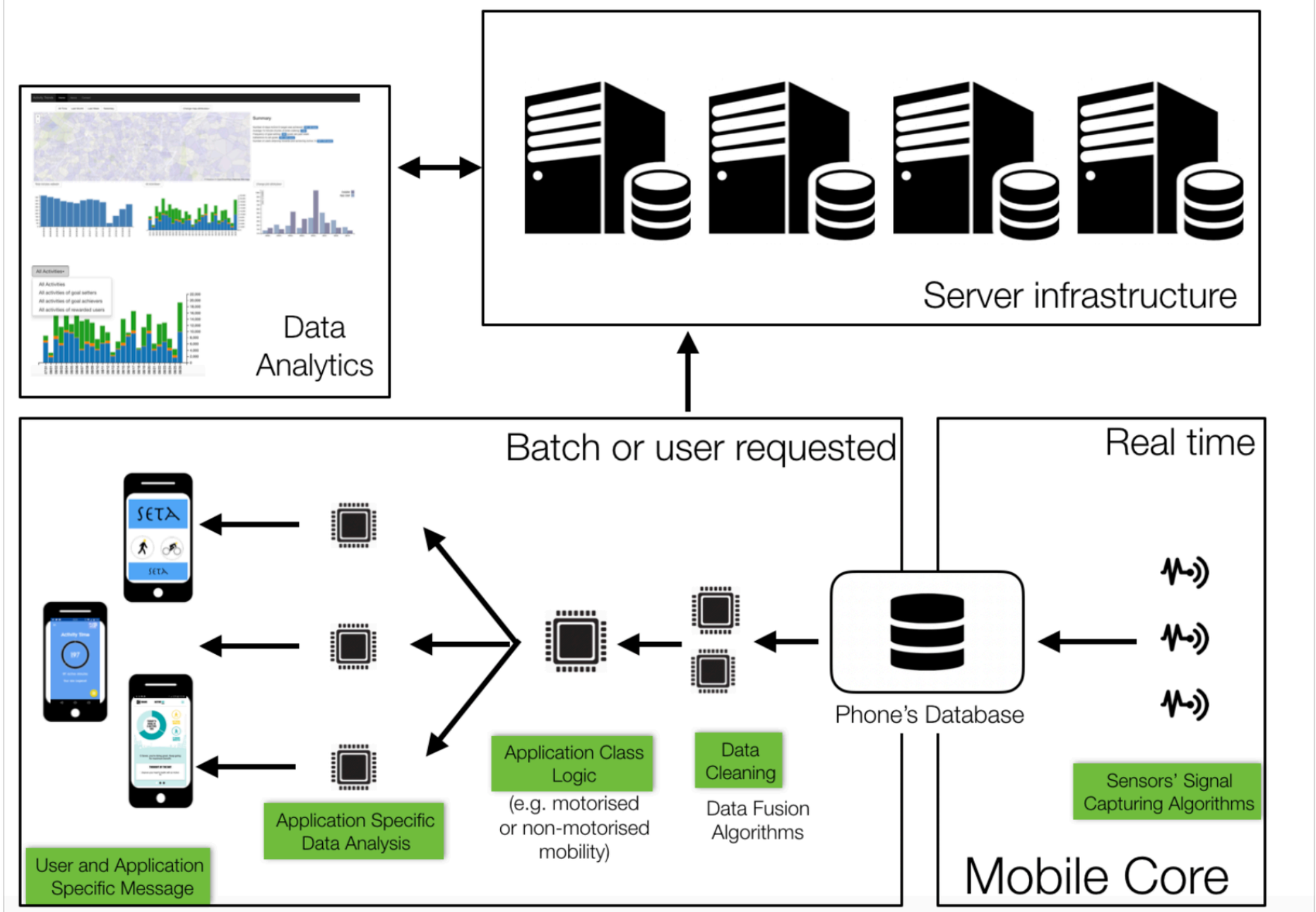
- **Tracking using mobile phones**
  - Using onboard sensors (mostly accelerometer based and location)
- **Requirements:**
  - Sub-minute level tracking of physical activity
  - Low battery consumption
  - Low data usage
- **Algorithms**
  - to collect data from sensors
  - to fuse the sensor data
  - to create geolocated multimodal journeys from sensors
- **Large server collection architecture able to scale up to millions of users and terabytes of data**

Mobile architecture

- **Algorithms** accessing, interpreting, fusing, contextualising, and storing the sensors' signal.
- The logic for **tracking applications**.
  - We can track different types of mobility such as motorised mobility or non-motorised mobility. The output is provided into a phone's internal database which will be used as means of communication between the modules
- **Application specific data analysis**.
  - Given a specific application (e.g. MoveMore), it is possible to further analyse the data to provide the specific app's user interface.
  - **Generic architecture for sensing non-motorised and motorised mobility**
  - 24/7, Battery preserving, data allowance preserving, working on dozens of thousands of phone models
- All data is sent to a **central cluster**
  - typically 2 servers: one for acquisition and one for processing – depending on load levels the two servers can be supported by one single virtual machine.
- The data on the server is retrievable and **analysable for data analytics purposes**.
  - We provide tools to analyse both population levels indicators and individual indicators (WP5)

The University Of Sheffield.

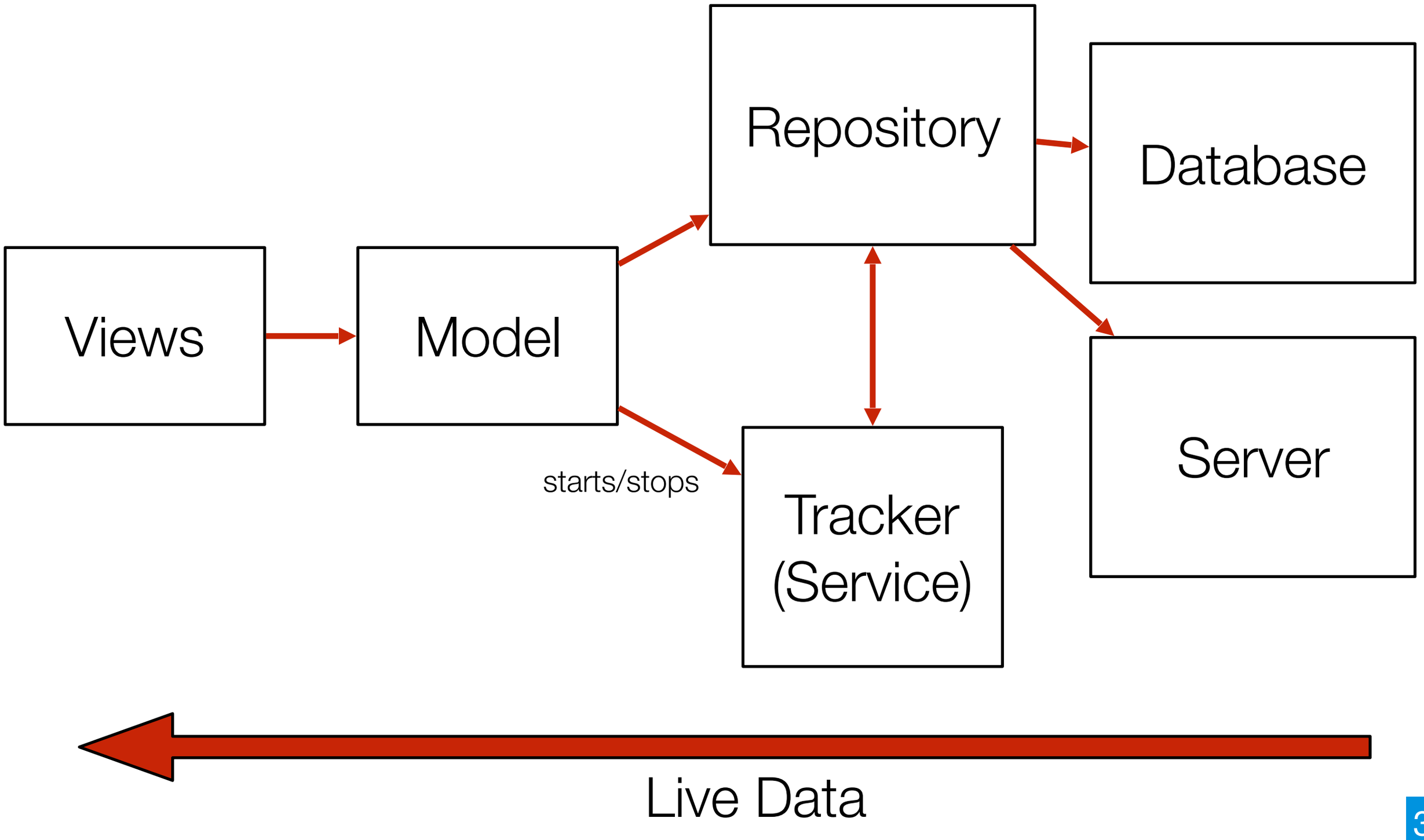
The Engineering



A simplified view

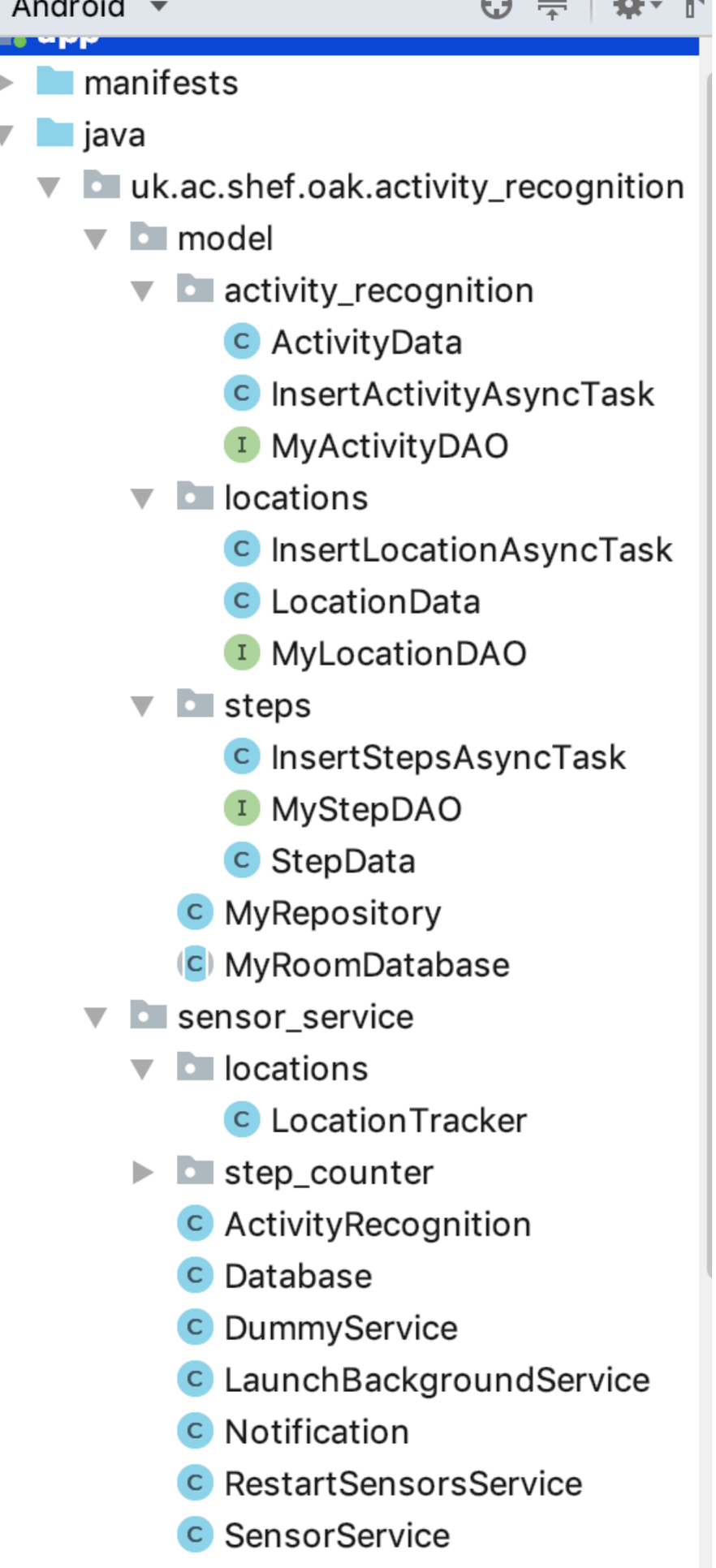
- Using the technologies we have seen this semester
  - Please note:
    - The real architecture is ***far* more complex**: this version would not work over large scale as several issues of incompatibility across vendors will make it fail on many phones
    - You will also need to implement a number of **tricks for special situations** (phone rebooted, Doze/App Idle mechanisms, service killed, Android 8+ foreground style, etc).
    - Also battery-wise it would be quite bad - you need to implement saving strategies (e.g. turning off locations when unneeded)
    - I am omitting several other sensors that improve quality and stability
    - You will need strategies to cope with failure to deliver of sensors (very common)
    - But it is a start

Organisation



Sensing and Storing data

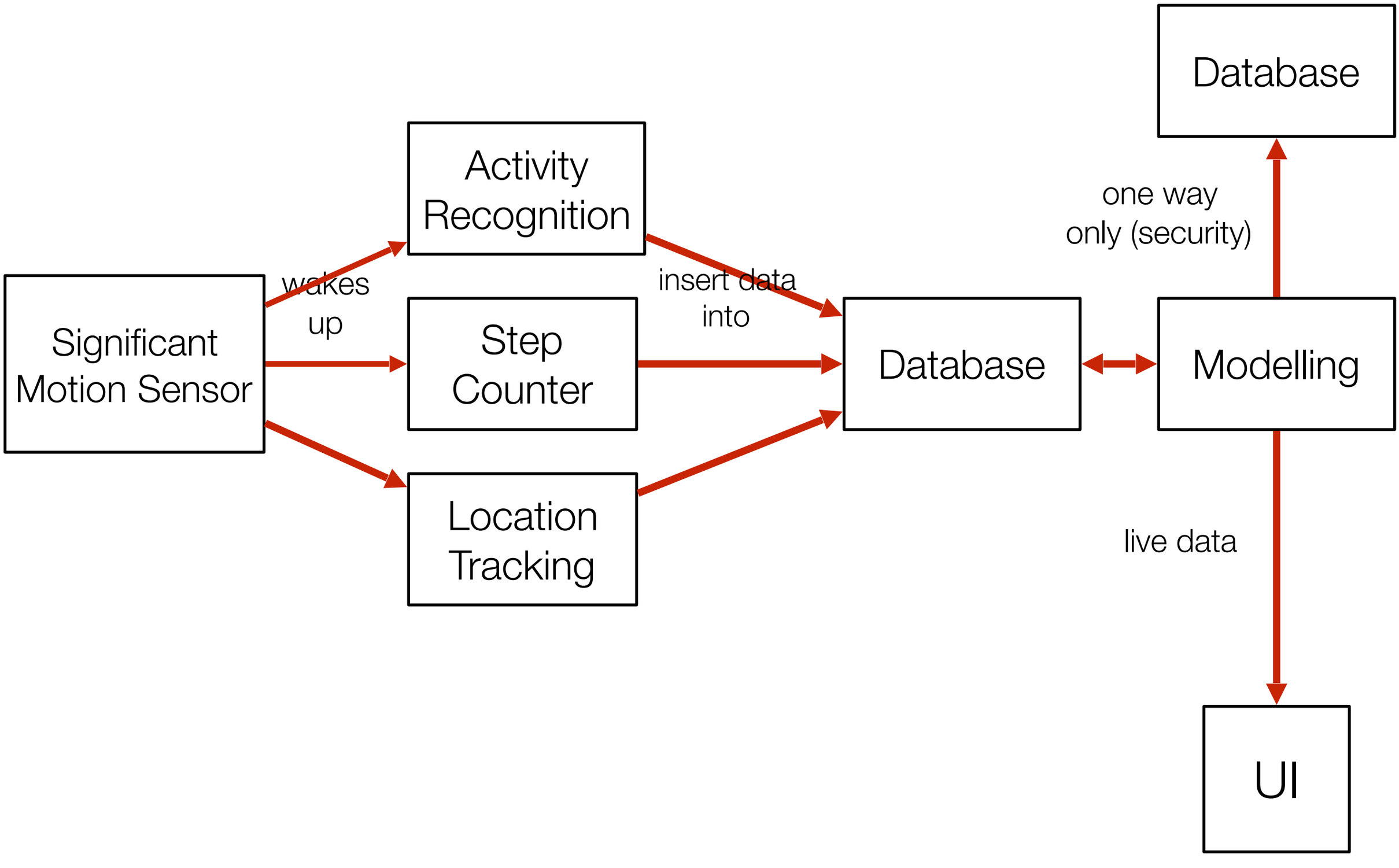
- The tracker runs in the background 24/7 independently from the lifecycle of the app
  - Installation phase sets all permissions and launches the background service which does the tracking
- The service is very small
    - otherwise it will use battery and resources (either user or Android would kill it)
    - it just receives data from the sensors and stores it into the database
    - it restarts is it is killed



- Database: three relations
  - implemented using Rooms
  - Activity Recognition
    - (recognising walking, cycling, etc.)
  - Locations (timed GPS coordinates)
  - Timed Steps
- Sensor Tracking Service:
  - Location tracking every 30 seconds
  - Activity recognition every time activities changes
  - Steps every 20 seconds

Simplified View

Tracking (simplified view)



Example Steps

```
import ...

@Entity()
public class StepData {
    @PrimaryKey(autoGenerate = true)
    @android.support.annotation.NonNull
    private int id=0;
    private int steps;
    private long time;
    private double speed;

    public StepData(int steps, long time, double speed) {
        this.steps = steps;
        this.time= time;
        this.speed= speed;
    }

    @android.support.annotation.NonNull
    public int getId() { return id; }
    public void setId(@android.support.annotation.NonNull int id)


    public int getSteps() { return steps; }

    public void setSteps(int steps) { this.steps = steps; }

    public long getTime() { return time; }

    public void setTime(long time) { this.time = time; }

    public double getSpeed() { return speed; }

    public void setSpeed(double speed) { this.speed = speed; }
```

The University Of Sheffield.

Steps DAO

```java
@Dao
public interface MyStepDAO {
    @Insert
    void insertAll(StepData... stepData);

    @Insert
    void insert(StepData stepData);

    @Delete
    void delete(StepData stepData);


    @Query("SELECT * FROM StepData where time= :time ")
    public LiveData<StepData> getStepsAt(long time);

    @Query("SELECT * FROM StepData where time>= :midnight and time<= (:midnight +86400000) ORDER BY time DESC LIMI
    public LiveData<StepData> getCurrentLastSteps(long midnight);


    @Query("SELECT * FROM StepData where time>= :time ORDER BY time ASC LIMIT 1")
    public LiveData<StepData> getStepsAfter(long time);



    // it selects a random element
    @Query("SELECT * FROM StepData where time>= :startTime and time<= :endTime ORDER BY time ASC")
    public List<StepData> getStepsBetween(long startTime, long endTime);

    @Delete
    public void deleteAll(StepData... stepData);

    @Query("SELECT COUNT(*) FROM StepData")
    public int howManyElements();
}
```

Screenshot

The University Of Sheffield.

Step Sensor

```
public StandardAndroidPedometer(Context context, Database databaseX, int mSecsFrequency) {
    super(context, databaseX, mSecsFrequency);
    if (isKitkatWithStepSensor(context)) {
        // http://androidforums.com/threads/how-to-get-time-of-last-system-boot.548661/
        timePhoneWasLastRebooted = System.currentTimeMillis() - SystemClock.elapsedRealtime();
        mSamplingRateNano = (long) (mSecsFrequency) * 1000000;
        mSamplingRateInMSecs = (long) mSecsFrequency;
        mSensorManager = (SensorManager) context.getSystemService(Context.SENSOR_SERVICE);

        mStepCounterSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_STEP_COUNTER);
        if (mStepCounterSensor == null) {
            Log.d(TAG, msg: "Standard Pedometer unavailable");
    } else {
        Log.d(TAG, msg: "Using Standard Pedometer");
        mStepCounterListener = new SensorEventListener() {
            @Override
            public void onSensorChanged(SensorEvent event) {
                long diff = event.timestamp - lastStepReportTime;
                // time is in nanoseconds it represents the set reference times the first time we come here
                // set event timestamp to current time in milliseconds
                // see answer 2 at http://stackoverflow.com/questions/5500765/accelerometer-sensorevent-tim
                if (diff >= mSamplingRateNano) {
                    long time = timePhoneWasLastRebooted + (long) (event.timestamp / 1000000.0);
                    int steps = (int) event.values[0];
                    insertStepsFromData(steps, time, speed: -1, delta: false);
                    lastStepReportTime = event.timestamp;
                }
            }

            @Override
            public void onAccuracyChanged(Sensor sensor, int accuracy) {
```

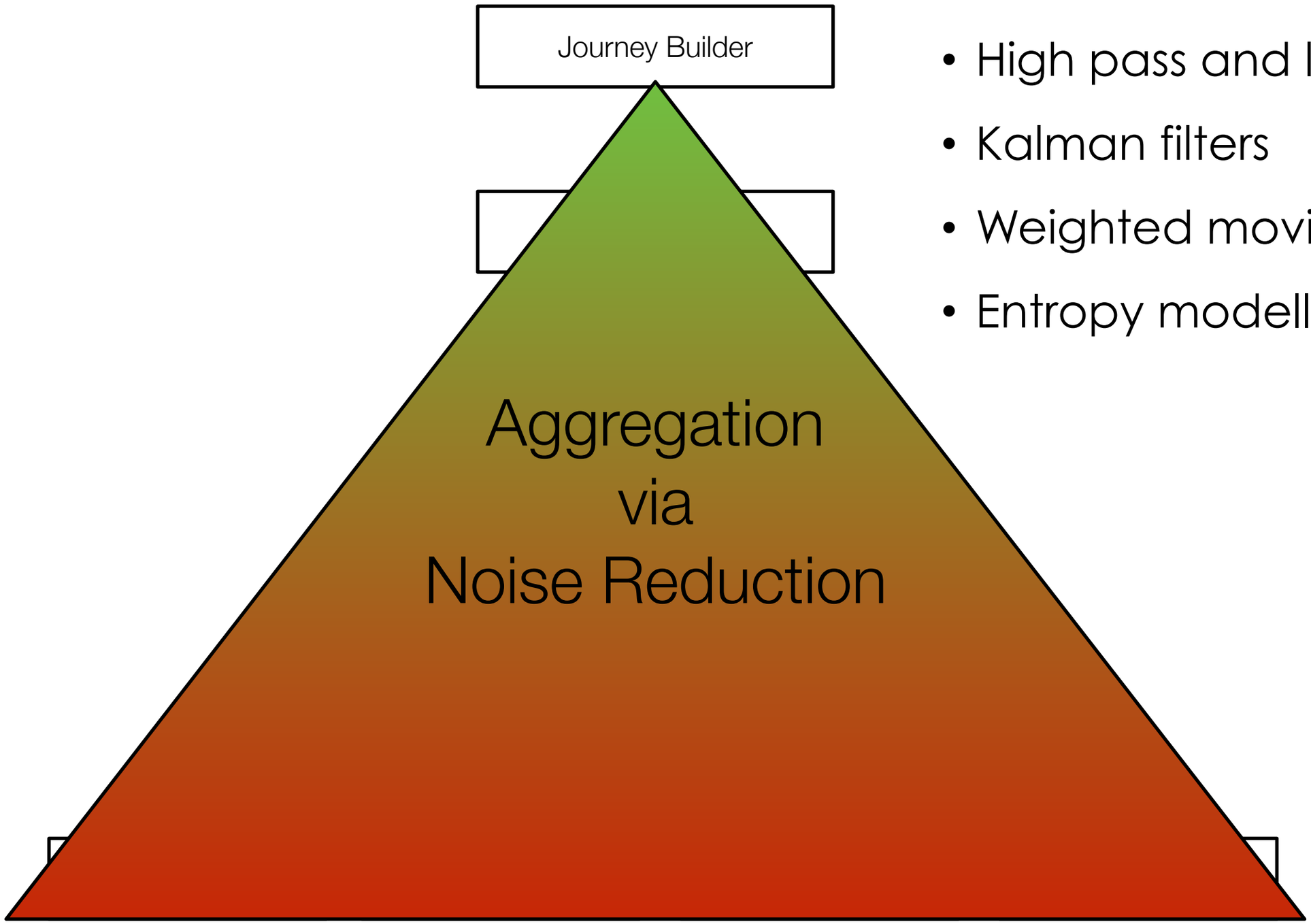
Screenshot

```
    public boolean startStepCounting() {
        super.startStepCounting();
        // if the sensor is null,then mSensorManager is null and we get a crash
        if (standardPedometerAvailable()) {
            Log.d( tag: "Standard StepCounter",  msg: "starting listener");
            // delay is in microseconds (1millisecond=1000 microseconds)
            // it does not seem to work though
            //stopStepCounting();
            // otherwise we stop immediately because
            mSensorManager.registerListener(mStepCounterListener, mStepCounterSensor, (int) (mSamplingRateIn
        }
        return true;
    }

    public void stopAndRestartPedometer() {
        stopStepCounting();
        startStepCounting();
    }

    public void stopStepCounting(){
        super.stopStepCounting();
        flush();
        try {
            mSensorManager.unregisterListener(mStepCounterListener);
        } catch (Exception e) {
            ;// probably already unregistered
        }
    }


    public void flush() {
        try{
            if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.KITKAT)
                mSensorManager.flush(mStepCounterListener);
        } catch (Exception e){
```

All the rest

- We saw the code for Activity Recognition Transitions and Locations last week
  - Just refer to that code
- **The model is the most complex part**
  - when you put together all the sensor readings for modelling mobility in details
    - e.g cadence
- All the rest is just Android specific complexity
  - **remove all errors** (e.g. misinterpretation of activities - vehicle interpreted as walking)
  - cope with **non standard misbehaviour** (e.g. sensors that return a flat value, services killed by Android without reasons, etc.)

Data analysis



- Classification algorithms
- High pass and low pass filters
- Kalman filters
- Weighted moving average
- Entropy modelling

Server

- Provides scalable and reliable backend for mobile applications
  - >24 million/day query
  - >1 million queries/hour
  - 0.247ms/12.6ms minimum/maximum response time
  - >30K/hour write throughput
- data safety and privacy are paramount

The University Of Sheffield.

A Never Ending Service

Android

- Never ending background processing
  - able to recover if user swipes out the app
  - https://fabcirablog.weebly.com/blog/creating-a-never-ending-background-service-in-android
- Foreground process to avoid Android stopping the process
  - it creates a permanent notification
    - user cannot swipe - needs hiding with a trick
- Regular resurrection of process to avoid
  - killing by Android
  - phone's deep sleep

The University Of Sheffield.

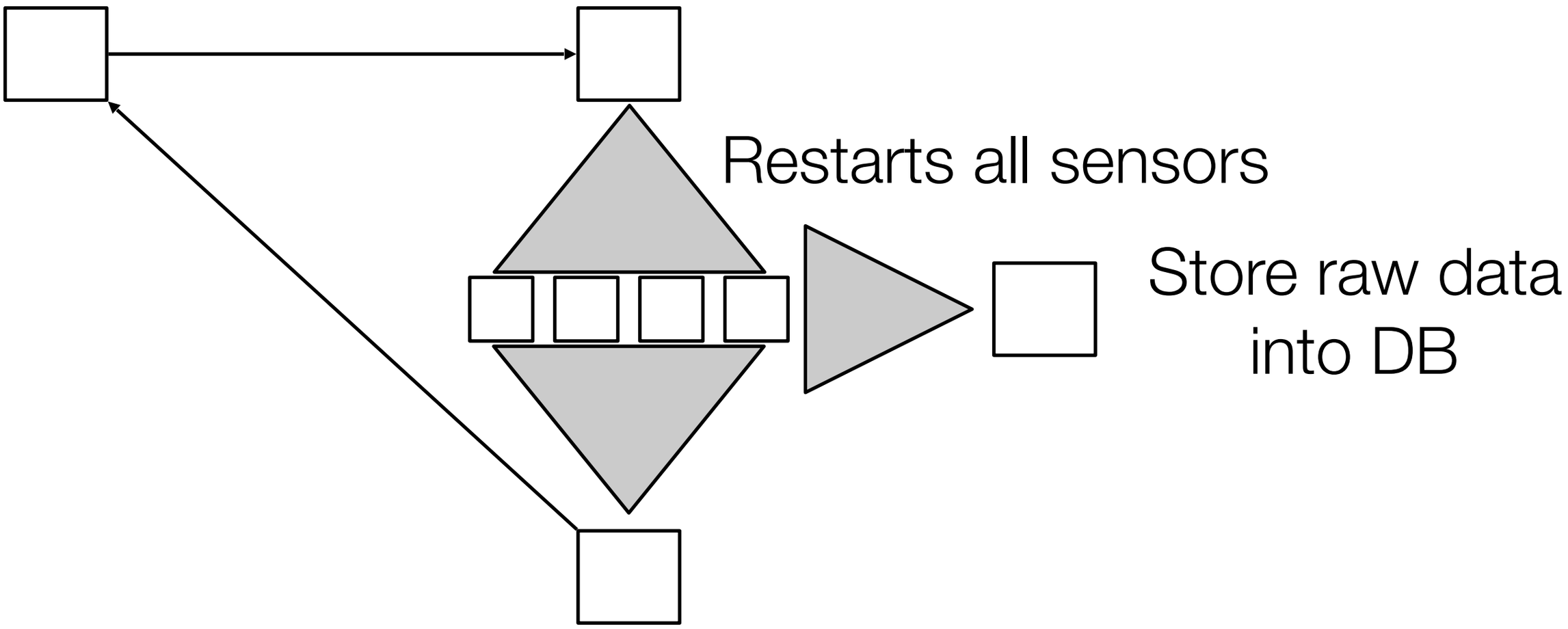
Saving battery

- The process cannot run constantly
- The phone needs to sleep
- Use the Significant Motion sensor
  - it will activate when the phone moves significantly
    - User grabs phone
    - User starts walking
    - User is in a car

The University Of Sheffield.

Tracker sleeps

Significant motion identified



Summary

- Part 1:
  - Releasing an app over large scale
- Part 2:
  - Examples of apps we have developed
- Lab tutorial :
  - Preparing assignments