School of Computer Science and Statistics

Tools for the Investigation of Physical Activity and its Effect on Software Engineering Performance

Ryan Dowling

April 19, 2022

Supervised By: Prof. Stephen Barrett

A dissertation submitted in partial fulfilment of the requirements for the degree of Master in Computer Science
Declaration

I hereby declare that this dissertation is entirely my own work and that it has not been submitted as an exercise for a degree at this or any other university.

I have read and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at http://www.tcd.ie/calendar.

I have also completed the Online Tutorial on avoiding plagiarism 'Ready Steady Write', located at http://tcd-ie.libguides.com/plagiarism/ready-steady-write.

Signed: ___________________________      Date: ___________________________
Abstract

It’s commonly said that doing physical activity after a long period of working can help to clear the mind thus helping it to focus and perform better.

This dissertation aims to develop tools to aid researchers in investigating whether or not this hypothesis is true, specifically focusing on if physical activity has any influence on software engineering performance.

Two tools were developed for this dissertation. The first is a mobile phone application which uses built-in device sensors to track the user’s movement and uploads the data to a database. The second tool is a web server and dashboard that pulls metrics to measure software engineering performance from the user’s Github profile and compares them with the user’s movement data.

An experiment was run on the author to gather their movement data and software engineering performance data over a short time period.

Overall the tools succeeded in collecting the required data to compare the physical activity and software engineering activity of a user. No correlation was found between the two over the period of the short experiment.

The outcome of this dissertation is that researchers in the future could use the tools created by this dissertation to run more experiments to perform a more accurate investigation into the hypothesis proposed above.
Acknowledgements

Firstly, I would like to thank my supervisor, Prof. Stephen Barrett for his tremendous support and guidance throughout the duration of the dissertation.

I would like to thank my friends for putting up with me for the past five years. Without their help and encouragement over the past few years I would not have made it to where I am today.

Finally, I would like to thank my family for their constant love and support, throughout my degree and my entire life and without whom none of this would be possible.
Contents

1 Introduction ................................................. 1
   1.1 Motivation ........................................... 1
       1.1.1 Improving Software Engineering Performance .......... 1
       1.1.2 Physical Health of Software Engineers .................. 1
   1.2 Research Question and Objectives ..................... 2
   1.3 Workflow ............................................ 2
   1.4 Overview of Dissertation ................................ 3

2 State of the Art ............................................ 4
   2.1 Physical Activity and Brain Cognition ................... 4
       2.1.1 Aerobic Capacity and Brain Cognition .................... 4
       2.1.2 Acute Exercise and Brain Cognition .................... 5
   2.2 Measuring Physical Activity ............................. 6
       2.2.1 Gathering Data with Mobile Devices ................... 6
   2.3 Measuring Software Engineering .......................... 6

3 Design ..................................................... 8
   3.1 Mobile Application ..................................... 8
       3.1.1 Mobile vs Wearable ..................................... 8
       3.1.2 Collected Data ....................................... 8
       3.1.3 Application Design ..................................... 9
   3.2 Dashboard ............................................. 11
       3.2.1 Collected Data ....................................... 11
       3.2.2 Application Design ..................................... 11

4 Implementation ........................................... 15
   4.1 Database ............................................. 15
   4.2 Mobile Application ..................................... 16
       4.2.1 Framework and Language ............................... 16
       4.2.2 Data Collected ....................................... 16
## 4.2.3 Login Screen .................................................. 16
## 4.2.4 Step Counter Screen ........................................ 17
## 4.3 Dashboard ....................................................... 19
## 4.3.1 Framework and Language .................................. 19
## 4.3.2 Accessing Github’s API .................................... 19
## 4.3.3 Login Page .................................................... 19
## 4.3.4 Main Page .................................................... 20

## 5 Evaluation ........................................................... 23
## 5.1 Evaluation of tools .............................................. 23
## 5.1.1 Mobile Application .......................................... 23
## 5.1.2 Dashboard .................................................... 23
## 5.2 Limitations and Unimplemented Features ..................... 24

## 6 Conclusions .......................................................... 26
## 6.1 Conclusion ....................................................... 26
## 6.2 Future Work ..................................................... 26

A1 Appendix ............................................................ 31

A2 Appendix ............................................................ 32

A3 Appendix ............................................................ 33
List of Figures

2.1 Left: Aerobic capacity vs. Mathematical Achievement in school-aged children. Right: Aerobic Capacity vs. Reading Achievement in school-aged children. (1) 4

2.2 Effect size of participants in experiments for brain cognition, including executive control, controlled processing, visiospatial and Speed tests (2) (1) . . . 5

3.1 Market Share of Smartphone Shipments (3) . . . . . . . . . . . . . . . . . . 9
3.2 Mobile Application Architecture . . . . . . . . . . . . . . . . . . . . . . . 10
3.3 Dashboard Architecture for Test Participant . . . . . . . . . . . . . . . . . 12
3.4 Dashboard Architecture for Researcher . . . . . . . . . . . . . . . . . . . . 13

4.1 Database Structure . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15
4.2 Login Screen for Mobile Application . . . . . . . . . . . . . . . . . . . . 16
4.3 Step Counter Screen for Mobile Application . . . . . . . . . . . . . . . . . 17
4.4 Axes read by acceleration sensor . . . . . . . . . . . . . . . . . . . . . . . 18
4.5 Login Page of Dashboard . . . . . . . . . . . . . . . . . . . . . . . . . . . 20
4.6 Main Page of Dashboard . . . . . . . . . . . . . . . . . . . . . . . . . . . . 22

5.1 Step Count (BLUE) and Commits (RED) of a Repository over time . . . . . . 25
5.2 Step Count vs. Commits of a Repository . . . . . . . . . . . . . . . . . . . 25
1 Introduction

This chapter will outline the motivation for the dissertation. It will also include the research question this dissertation will be addressing and the objectives of this dissertation, as well as an overview of the following chapters that will be included in this paper.

1.1 Motivation

This section will outline the motivation behind the topic of this dissertation.

1.1.1 Improving Software Engineering Performance

There have been many studies performed surrounding physical activity and its effect on cognitive brain function. Many of these, some of which will be discussed in Chapter 2, have shown a positive correlation between the two. However many of the tests for brain cognition performed, such as the Sudoku puzzle performed by researchers (4) are not specific to the field of software engineering.

One of the main motivators of this dissertation is to find a direct link between the physical activity and the performance output of a software engineer. Examples of performance outputs will be discussed in more detail in Chapter 2. The ideal outcome for future researchers using these tools would be a positive correlation between physical activity and software engineering performance, thus encouraging more software engineers to become physically active and improve their engineering performance.

1.1.2 Physical Health of Software Engineers

Much of the work performed by a typical software engineer, be it programming or in meetings, involves sitting down for prolonged periods of time. Studies have shown that there are numerous health risks associated with prolonged sitting, such as in a study published in The Spine Journal (5), which shows an increased risk of spinal injury after performing full flexion movements in a person after periods of prolonged sitting. Due to the nature of software engineers’ work, this puts them at risk of these types of injuries.
As mentioned in the previous subsection, an ideal outcome for future researchers using these tools would for there to be a positive correlation between physical activity and software engineering performance. In finding such a correlation, more software engineers might be motivated to become more physically active, which would have physical health benefits to coincide with the engineering performance benefits mentioned previously.

1.2 Research Question and Objectives

The initial aim of this dissertation was to answer the following research question:

RQ: Is there a correlation between doing physical activity and software engineering performance?

Due to the scope of this project, this question was unable to be answered in its entirety. Thus, the topic of this dissertation shifted to answer the following research question:

RQ: Can tools be built to automate the process of measuring physical activity and software engineering performance?

The objective of this dissertation is to answer this research question by building two tools, a mobile phone application for measuring physical activity, and a web dashboard that will measure software engineering performance, as well as displaying the two datasets graphically. The aim is that researchers in the future can pick up the tools created for this dissertation to give them a foundation to continue the investigation of the original research question proposed.

1.3 Workflow

Prior to the beginning of work on this dissertation, key stages were planned as part of the workflow. These stages include:

- Defining the research question
- Background research on physical activity and brain cognition, measuring physical activity and software engineering metrics
- Designing and creating the mobile application
- Collecting of data using the mobile application
- Designing and creating the dashboard
- Evaluating the tools
1.4 Overview of Dissertation

The remainder of this dissertation will be divided into the following chapters:

- State of the Art will explain the background details surrounding previous research done on physical activity and cognitive brain function. It will also include information on ways physical activity can be measured as well as information on software metrics.

- Design will outline the design approach of the tools created for this dissertation, namely the mobile application for measuring physical activity, as well as the dashboard for retrieving engineering data and comparing it with the engineering data.

- Implementation will outline the implementation of the mobile application and dashboard based on their designs in the previous chapter.

- Evaluation will outline the results of the tools implemented in the previous chapter and their relation to the research goal, as well as some findings from the results.

- Conclusion will provide the conclusion for the dissertation, including an overview of the contributions made as well as potential work that may be done in the future.
2 State of the Art

This chapter contains an overview of the research done surrounding the research questions stated in Section 1.2. Section 2.1 contains an outline of existing research done on the topic of physical activity and its effect on brain cognition. Section 2.2 will contain an outline of the types of physical activity that can be measured, and the devices currently available to do so. Section 2.3 contains an outline of the types of metrics that can be gathered from software engineering related activities.

2.1 Physical Activity and Brain Cognition

2.1.1 Aerobic Capacity and Brain Cognition

The effects of physical activity or exercise on brain cognition is an area of research that has long been under investigation. Much of the results point towards a positive correlation between, i.e. performing exercise has positive impacts on the brain. Many studies performed in this area are based on the individual’s capacity to do exercise compared to their ability to perform cognitive tasks. We can see an example of this in a study performed by JH Martin (6). The author tested the aerobic capacity of school-aged children by way of a PACER (7) test versus their abilities in both mathematical and reading achievements. A small, but nevertheless positive correlation was shown, as shown in Figure 2.1.

![Figure 2.1: Left: Aerobic capacity vs. Mathematical Achievement in school-aged children. Right: Aerobic Capacity vs. Reading Achievement in school-aged children. (1)](image)

This correlation is not only seen in young children, but in older aged adults too. In another
study performed by Colcombe and Kramer (2), an experiment was run on older aged adults (ages ranging from 55-80 years old) in which the participants performed tests for brain cognition. The participants were split between a control group of participants who did not exercise, and a group that took part in aerobic fitness training groups. As seen in Figure 2.2, the participants in the exercise groups showed a much larger effect size in the cognitive experiments than those who did not.

Figure 2.2: Effect size of participants in experiments for brain cognition, including executive control, controlled processing, visiospacial and Speed tests (2) (1)

2.1.2 Acute Exercise and Brain Cognition

These previous two studies show a positive correlation between physical activity and brain cognition, however, these are primarily concerned with how an individual’s capacity to perform physical activity has an effect on their brain cognition. The tools that are built for this dissertation are to test whether the act of doing physical activity has an effect on a person’s engineering performance, e.g. going for a run before starting to code. This is known as acute aerobic exercise. While not specific to software engineering, research has been done to test whether or not doing physical activity has an effect on cognitive thinking. In study by Chang et al. (8), participants were asked to perform the Tower of London (9) task, with one group doing 30 minutes of exercise prior to attempting it. It was shown that those in the exercise group had benefits to their executive functions of planning and problem solving when completing the task. This again shows that physical activity has a positive correlation with brain cognition.
2.2 Measuring Physical Activity

Physical activity is any bodily movement that requires energy to be expended in order for it to be performed (10). In the previous section, physical activity was primarily discussed in terms of aerobic exercise, however this is not the full extent of what physical activity is. Actions like standing up or walking around the office also constitute as performing physical activity. As there are numerous types of physical activity, there are numerous measurements associated with these activities. Such examples include step count, running speed or VO\textsubscript{2} \textit{max} (11).

In terms of gathering physical activity data, there are numerous technologies available to perform these tasks. The most common devices available to gather physical activity data are smartphones and wearable devices. It’s estimated that there are currently 6.64 billion smartphone users (12), including devices produced by Samsung, Apple and Nokia. Wearable devices, such as Apple Watches or Fitbits, are also very popular, with an estimated 722 million devices worldwide (13). Mobile devices were chosen to be investigated further, for reasons that are explained in Chapter 3.

2.2.1 Gathering Data with Mobile Devices

When gathering data with mobile devices, built in sensors are the typical way this is done. These types of sensors include accelerometers, gyroscopes and thermometers to name a few (14). In a paper by Ferreira, D., et al. (15), a toolkit was created to gather sensor data from Android mobile devices. A framework was created that, at the time of the paper’s release, could be used in any Android application. For research purposes, an application called the Aware client was also created. This application contained plugins to support more than the typical pre-installed sensors. This application stored data collected from the sensors by default to the device, but was able to upload the data to a database too. It was also optimised to reduce energy consumption, which is particularly useful over the duration of a study where participants would want to constantly be gathering data.

2.3 Measuring Software Engineering

Since the inception of software development, as far back as the 1960s, developers have been interested in the measurement of code properties. These early metrics could be used as a basic form of measurement of a software engineer’s performance. Some examples of metrics include Lines of Code (LOC) in a given time period, or bugs per n LOC (16). These metrics can be used as a simple way to measure the productivity of an engineer, which in turn can be used to compare engineers against industry standards or against other engineers.
To this day, metrics are still used to measure the productivity of engineers. Naturally, as
time as progressed, more and more ways of measuring software engineering have been
created. Examples of these include the amount of bugs in a code base compare to the lines
of code, or the effort it takes to fix code issues compared to the total time allotted to the
project (17).

Improvements to software engineering metrics and their results have been investigated by
researchers in recent years. A study performed by Yue Jian, et al (18). investigated methods
for improving software quality predictions. In this study, supervised machine learning was
deployed to focus on improving the information content of their training data. The results of
their research showed evidence that the choice of the right metrics showed the largest
difference in the quality of predictions, rather than deploying other machine learning
algorithms.
3 Design

This chapter will outline the design details and decisions behind the tools created for this dissertation. Section 3.1 will outline the details of the mobile application, used for measuring physical activity and Section 3.2 will outline the details of the dashboard, used for measuring and displaying the software engineering metrics.

3.1 Mobile Application

3.1.1 Mobile vs Wearable

As mentioned in Section 2.2, both mobile and wearable devices are two of the most common ways of gathering physical activity data. For this dissertation mobile devices were chosen as the platform to create the physical activity data gathering tool. The reasoning behind this decision is for ease of use in future research work.

As mobile devices are more popular, it will make future research easier by way of more people having access to mobile devices than wearable devices, therefore making future test participants easier to recruit.

In terms of technical reasons, mobile devices have more library support than the wearable devices. On top of this, programming for mobile devices means easier cross platform integration and generally have an easier set up, which will be convenient for researchers using the tools. Wearable device applications are typically programmed specifically for the wearable devices themselves, such as Garmin device applications, which use their own programming language, Monkey C (19).

3.1.2 Collected Data

In Section 2.2, examples were given of types of physical activity data that could be tracked. For this dissertation, step count was the data chosen to be measured. Many of the studies mentioned in previous sections have used VO₂ max as the physical activity measurement to track. The reasoning behind choosing step count over VO₂ max is that VO₂ max requires
the user to actively track the measurement with a mobile device, while step count can be done passively as the user goes about their day.

### 3.1.3 Application Design

As this tool will be used for research purposes, the mobile application was designed to be easy to use for a test participant to track their steps, without having to play around with multiple settings to get it up and running.

#### Operating System and Framework

The mobile application created for this dissertation was programmed in JavaScript using the React Native (20) software framework. This framework allows for quick set up, as well as simple cross platform integration, without having to code specifically for both Android and iOS.

Despite the cross platform capabilities available from React Native, this dissertation was tested primarily on Android devices, both emulators and physical devices. Due to the scope of this dissertation, Android devices were chosen due to their availability, as it might be beneficial for future researchers to have tools working on a more popular platform. The breakdown of the market share of mobile devices can be seen in Figure 3.1.

---

**Figure 3.1: Market Share of Smartphone Shipments (3)**
Mobile Architecture

Figure 3.2 gives an overview of how the Android application will work. While the application is running on the device, the user’s step count will be collected by built-in sensors in the device. In order to track steps, the accelerometer sensor will be used. While there are step counter sensors available, not all Android devices have these built in, while all have an accelerometer sensor. If development of these tools for iOS devices is ever done in the future, this will also be useful as iOS devices use their accelerometer sensors for calculating steps too.

In order to differentiate test participants, a login page will appear allowing them to enter their Github username. This username will be tested with Github’s API, and provided it is valid will allow the user to proceed to the main app.

On the main app, when a step is recorded, two things will happen. Firstly, the running count of steps on the screen will update to reflect the new step. Secondly, the new data will be stored both locally as well as in a database. Local storage is used in case of internet connectivity issues, allowing data to be recovered in case of the user closing the app and reopening in a time when they cannot connect to the database.

The choice of database for this dissertation is a Firebase Realtime database (21). This type of database was chosen for multiple reasons. The first is that it works in real time, which allows the data to be constantly updated (provided an internet connection is available) and not requiring the user to manually upload their data at the end of a physical activity session. The second reason for this choice of database is that it easily integrates with the web server and dashboard, which will be discussed in Section 3.2.

![Figure 3.2: Mobile Application Architecture](image)

Security

When dealing with personal data, there are security and privacy concerns to consider with regards to how this data is stored. This tool collects step count data, which while not the most private, is still considered personal data and thus must be treated with sensitivity.
Having privacy features implemented will also be a boon to potential future works on the tool, if say other forms of physical activity data were to be recorded in the future.

Taking these things into consideration, the data collected from the device sensors will be encrypted prior to it being pushed to the real-time database, where it will be available to be decrypted later on by the web server.

A simple form of AES encryption will be used on the step counter data prior to it being added to the database.

### 3.2 Dashboard

#### 3.2.1 Collected Data

Like with the mobile application, data had to be selected to be used as a measurement of software engineering performance. In a similar vein to the previous tool, it was decided to keep the data collected simplistic, and for this reason the metrics of Commits, Additions of Lines of Code and Deletions of Lines of Code were chosen as the metrics to be measured.

All of these metrics are conveniently provided by Github’s API service (22), which was used extensively in this tool.

#### 3.2.2 Application Design

As with the previous tool, this dashboard tool was designed to be used for research purposes and thus was designed to be easy to use for both test participants and researchers, both of who will be using this dashboard.

The dashboard tool was created to both collect and display the software engineering metrics gathered, as well as to display these metrics alongside the physical activity data for comparisons.

This tool is to be used by both test participants, who will use it to see their data as well as upload their engineering data to the database and also be used by researchers, who can use the tool to see the data collected by the test participants for comparisons in their experiments.

#### Framework

Similar to the mobile application a framework was also used to build the dashboard that displays the collected data. Angular (23) was the choice of framework for this tool. While
other frameworks like React or Vue are available, the author had previous experience working with Angular which made it more feasible to use for this dissertation.

Authentication

For both types of users, test participants and researchers, authentication will be required to differentiate the two. A login screen will be created to allow this feature. Researchers will be able to login with specific admin credentials, while test participants will login with their Github credentials. After the user has signed in, they will be directed to their respective dashboard.

User Architecture

![Dashboard Architecture](image)

Figure 3.3: Dashboard Architecture for Test Participant

Figure 3.3 shows an overview of the architecture for the tool as used by a test participant. Two types of data will be displayed on this dashboard. The first is the user’s movement data as collected by the mobile application. This will be retrieved from the Firebase realtime database.

The second type of data collected will be the user’s software engineering performance data. This data will be retrieved from Github’s API, by retrieving data on the user’s repositories.
Repository data will be parsed for relevant coding metrics, as described in Section 3.2.1.

As of November 2020, Github no longer accepts username and password as a form of authentication for their API (24). For this reason, a Personal Access Token (PAT) will be required to retrieve the user’s repository data. User’s will have to generate a PAT on their Github profiles (25) to be used with the dashboard for retrieving their repository data.

Once the data has been retrieved, it will be displayed graphically alongside the user’s movement data for them to compare. The dashboard will also upload the data to the Firebase realtime database for use by the researchers.

Admin Architecture

Figure 3.4 shows an overview of the architecture for the tool as used by a researcher. Like the dashboard for test participants, the researcher dashboard will graphically display both movement data and software engineering performance data for comparisons against one another. However, the researcher differs in that it will show the data of all test participants, rather than just a single user.

Differing from the test participant dashboard again, the admin dashboard will not use Github’s API as all the data required will have been stored in the database.

![Figure 3.4: Dashboard Architecture for Researcher](image)
Security

Similar to the physical activity data collected, there are security and privacy concerns to be considered with the software engineering performance data being collected.

Personal Access Tokens will be used by the test participants to retrieve performance data from their respective repositories. As this only has to be done as the data is being retrieved, the access tokens will not be stored anywhere in the database.

Once concern that may arise is that repository data retrieved will include both public and private repositories. For this reason, it is extremely important that the data retrieved is secure.

As with the mobile app, data will be encrypted using AES before being pushed to the database, which can be decrypted on the dashboard again by the researchers when they will be reviewing the data.
4 Implementation

This chapter will outline the implementation of the tools, as described by the designs in Chapter 3. Section 4.1 will outline the implementation of the database, Section 4.2 will outline the implementation of the mobile application and Section 4.2 will outline the implementation of the dashboard.

4.1 Database

As mentioned in Chapter 3, a Firebase Realtime database is used as the primary data store for the data collected by both tools. The file structure for the database can be seen in Figure 4.1.

Figure 4.1: Database Structure
The config file created for the database (found on Firebase) is inserted in both the mobile application tool and dashboard tool, allowing the data to be pushed directly to the database.

4.2 Mobile Application

4.2.1 Framework and Language

As described in Section 3.1, a mobile application tool for Android devices was implemented to measure physical activity of a user. React Native was chosen as the framework to develop the application, which is programmed in JavaScript. CSS styling is used within the framework to style the application.

4.2.2 Data Collected

As described in Section 3.1.2, step count for a user is the measurement of physical activity being measured by this application. More specifically, this application is built to track the daily step count of the user.

4.2.3 Login Screen

The first screen the user is presented with is the login screen, as displayed in Figure 4.2. This screen was implemented with a simplistic design, simply requiring the user to enter their Github username.

![Figure 4.2: Login Screen for Mobile Application](image)
Authentication

When the user enters their username, a request will be made to Github’s API with the username using the endpoint `https://api.github.com/users/(username)` endpoint. One of three things will occur based on the response.

- User found: The user will be logged in and navigated to the next screen
- User not found: An error message will appear on the login screen notifying the user that the username entered does not exist
- Other errors: Any other error, for example a connection error, will appear on screen with a general error message

Once a user is logged in, the username will be saved locally to the app and they will not have to log in again until they log out.

4.2.4 Step Counter Screen

The second screen the user is presented with is the step counter screen, displayed in Figure 4.3. As with the first screen, this screen was implemented with a simplistic design, displaying the current daily step count, the username that the user is logged in with, and finally a way for the user to log out.

![Step Counter Screen for Mobile Application](image)

Figure 4.3: Step Counter Screen for Mobile Application

Sensor data

The step count data collected using this tool is gathered by using the accelerometer sensor of the device. As mentioned in Section 3.1.3, the accelerometer sensor was chosen due to its availability on all Android smartphone devices, whereas the pre-built step counter sensor is only available on newer models.
Every 100ms, acceleration along the x, y and z axes are calculated by the accelerometer sensor. The application finds the magnitude of this acceleration under the equation:

\[
\text{Magnitude} = \sqrt{x^2 + y^2 + z^2}
\]  

(1)

This magnitude is then subtracted from the previous 100ms’ magnitude to calculate a magnitude delta. If this magnitude delta exceeds the threshold set for a step, then a step is considered to have occurred and the relevant values are updated.

These steps can be tracked while the app is in the background which allows the data to be collected passively by the user as they are moving.

As the steps are updated, a running tracker on the screen increases displaying the current steps of the user for the day.

![Axes read by acceleration sensor](image)

Figure 4.4: Axes read by acceleration sensor

**Storing the data**

Once a step is recorded by the sensor, it is stored in two different ways, locally and to the Firebase Realtime database.

Locally, steps are stored every time the step counter is updated to keep a constant track of the latest values. This local data is also used on launch of the application, where it is pulled from local storage and set as the current step count. If no data exists, the step count is set to 0.

Data is pushed to the Firebase database using a library (26) for Firebase integration into React Native. Due to potential connectivity issues, as well as potential crashes, values are not stored in the database every time the step count is updated like the local storage. Instead, every 5000ms, data is taken from local storage and pushed to the Firebase database under \textit{Users/(username)/PhysicalActivityData/StepCount}.  

18
4.3 Dashboard

4.3.1 Framework and Language

As described in Section 3.2, a dashboard tool was implemented to measure software engineering performance of a user, as well as for displaying the two types of data collected against one another. This is for use of both test participants, who will use it to upload and view their engineering data, as well as researchers for reviewing the results.

Angular was chosen as the framework to develop the dashboard, which is programmed in TypeScript, along with a combination of HTML and CSS.

4.3.2 Accessing Github’s API

In order to access Github’s API, some form of authentication is required. As mentioned in Section 3.2.2, a Personal Access Token (PAT) is required to authenticate the user. The user will be required to generate their own token on their Github profile.

With the required authentication, this dashboard uses Octokit (27), a JavaScript library built for using Github’s API, to access and retrieve the required data from Github.

4.3.3 Login Page

The first page of the dashboard is the login page, which allows the user to login in with their Github credentials, as displayed in Figure 4.5.

Authentication

When the user’s credentials, username and PAT, are entered, a request using Octokit will be made to verify the PAT is valid. One of three things will happen after a response is received.

- Username and Token correct: The user will be logged in and navigated to the next page
- Username Incorrect and Token Correct: An error message will appear on the login page notifying the user that their token does not belong to their username
Other errors: Any other error, for example a token being incorrect, will appear on screen with a general error message.

4.3.4 Main Page

Retrieving Data

When the user signs in, data will be fetched from both Github and Firebase to retrieve their engineering data and movement data.

For the engineering data, Octokit will retrieve information from all the user’s repositories, using the https://api.github.com/users/username/repos endpoint. As a PAT has been used for authentication, this will retrieve information on both public and private repositories.

Once the data from all the repositories has been retrieved, the data is parsed through to pull out the relevant metrics that will be displayed. This includes data from all branches of all the repositories.

For the movement data, the data is retrieved from the Firebase realtime database. A library called AngularFire, a library that provides Firebase integration for Angular, is used to retrieve the signed in user’s movement data.

Both the data types are not collected in real time, and instead are collected upon launch of the dashboard. For this reason, a page refresh is required to update the data for the user if they wish to see new data that has been created.
Storing Data

When the Github data is retrieved, as mentioned in the previous subsection, it is automatically uploaded to the Firebase realtime database using AngularFire. The data is stored under Users/(username)/EngineeringData

Dashboard Layout

The dashboard has a simplistic layout, displaying two dropdown menus and two graphs. The layout can be seen in Figure 4.6. It was chosen to keep this simplistic as test participants would not want to mess around with many setting to be able to see their data.

Two graphs are created using D3.js (28), a graphing library for JavaScript. While Angular is programmed in TypeScript, D3 still works and is an effective way of graphing the data.

The first graph is a time series chart of the user’s daily steps to a given engineering performance metric over a period of time. The second graph is a scatter plot of the user’s daily steps vs. a given engineering performance metric.

The two dropdowns allow the user to cycle through the repository they wish to view the data of, as well as the engineering performance metric they wish to see on the graphs.
Visualization of Activity and Productivity Data

Repository: 
Metric: 

Physical Activity and Engineering Activity over Time

Physical Activity vs Engineering Activity

Figure 4.6: Main Page of Dashboard
5 Evaluation

This chapter will be an evaluation of the work produced for this dissertation. Section 5.1 will outline the tools outputs and their relation to the research question proposed. Section 5.2 will outline the limitations of these tools and the remaining work required for the tools to be fully functional for future research.

5.1 Evaluation of tools

5.1.1 Mobile Application

The mobile application built for this dissertation was created for the purpose of collecting physical activity data. No experiments on test participants were run during this dissertation, however a miniature experiment, in which the author used the tool to collect daily step count data for a trial period of time to verify that it was fully functional. Appendix C contains the data collected by the application in JSON format.

Due to constant configuration of the application and its sensitivity, some days have excessively large step counts while others have excessively small step counts. Despite this, the tool successfully recorded the physical activity of the user over a period of time.

This tool could be used by future researchers in tracking the physical activity of test participants who have it installed on their smartphones, which was one of the objectives of this dissertation.

5.1.2 Dashboard

The dashboard tool built for this dissertation was created for the purpose of collecting software engineering performance data, as well as graphically displaying these results on screen alongside the physical activity data collected.

Figures 5.1 and 5.2 show examples of the types of graphs generated by the dashboard.

Data displayed in these graphs was the movement data collected, measured in Section 5.1.1, as well as engineering performance data collected from the repository the author was using.
as a code base for the tools.

While not a full experiment, some early observations can be made from the graph. Much of the studies researched for this dissertation suggested a positive correlation between physical activity and engineering performance. The data produced however seems to be random and shows no strong correlation. In fact, as observed in the graph, there are many days with high step counts and no commits and days where the inverse is true.

While this might suggest that the research question initially proposed is false, there are multiple factors that influence the results produced in these graphs. Firstly, the period of recording data was very brief, and there are large sections of time where no data was recorded at all. Secondly, days where the author was programming tending to be days in which little physical activity is done, whereas days where lots of physical activity was being done would mean the author being away from the computer, and therefore little coding was done.

For these reasons, it’s difficult to draw conclusive results from the data produced my the small experiment. However, it can be seen that dashboard successfully collects and displays software engineering performance data, as per the objectives set out by this dissertation.

5.2 Limitations and Unimplemented Features

Despite the implementations of both of these tools, there are some limitations that may be required to be addressed before properly using these tools for future research. As well as this, there are some features that were specified in the design of the tools (Chapter 3) that did not make it in to the final implementation as they were outside of the scope of this dissertation.

One limitation of the tools created is the type of physical activity data collected by the mobile application. While step count is a sufficient measurement of physical activity, the problem lies in the time period over which it is collected. Currently, the tools only support the daily step count. As the initial research question is concerned with how physical activity directly influences software engineering performance, researchers would most likely want a more detailed breakdown of the participant’s step count throughout the day. The reason for this is they would be able to see specific time periods in which the user was more physically active, and see how their engineering performance within that time is influenced.

Another limitation, and feature that was unimplemented, was the researcher / administrator version of the dashboard. This dashboard would have allowed researchers to view the data of all users graphically, but due to it being out of scope of the dissertation this feature was not implemented. However, researchers would still have access to the Firebase database where
the data is stored, so they still have the option of reviewing test participants data.

The final unimplemented feature was the security feature of the data. Currently, both the mobile application and dashboard store unencrypted data collected, which may be of issue in the future provided the test participants wish for their data to be anonymised.

![Physical Activity and Engineering Activity over Time](image1)

**Figure 5.1:** Step Count (BLUE) and Commits (RED) of a Repository over time

![Physical Activity vs Engineering Activity](image2)

**Figure 5.2:** Step Count vs. Commits of a Repository
6 Conclusions

This chapter provides a conclusion to the dissertation, as well as direction for potential future work, both with the tools and on the tools themselves.

6.1 Conclusion

This dissertation initially set out to investigate whether or not there was any correlation between performing physical activity and a person’s software engineering performance. Due to scope changes, the project shifted towards instead building tools to support this investigation, so that researchers in the future would have a foundation to continue the investigation.

Over the course of this dissertation, two tools designed and implemented, a mobile application for gathering the physical activity data, and a dashboard for generating and displaying engineering performance data.

All in all the two tools produced were successful in answering the question of if tools can be built to automate the process of gathering physical activity and engineering performance data. With a few tweaks and additions, these tools will be full functional for future researchers.

6.2 Future Work

With the conclusion of this paper, there are multiple avenues in which future research can be taken:

- Implementing Remaining Features: As described in Section 5.2, there are still a few features remaining to be implemented before these tools can become fully functional for future research.
- Investigating the initial research question: This dissertation had originally set out to investigate whether or not there was a correlation between performing physical activity and software engineering performance. With tools now built to aid in this research,
future researchers now have a starting point to collect data which will hopefully aid in their investigations.
Bibliography


::text=Smartphones%20running%20the%20Android%20operating,percent%20share%20of%20the%20market.


A1 Appendix

Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>P.A.C.E.R.</td>
<td>Progressive Aerobic Cardiovascular Endurance Run</td>
</tr>
<tr>
<td>AES</td>
<td>Advanced Encryption Standard</td>
</tr>
<tr>
<td>CSS</td>
<td>Cascading Style Sheets</td>
</tr>
<tr>
<td>HTML</td>
<td>HyperText Markup Language</td>
</tr>
</tbody>
</table>
A2 Appendix

Source Code

- Mobile Application code: https://github.com/dowlinry/MCS/tree/main/ActivityTracker
- Dashboard code: https://github.com/dowlinry/MCS/tree/main/ProductivityTracker
A3 Appendix

Experiment Data

- Experiment Data: https://github.com/dowlinry/MCS/ExperimentDataResults.json