

Faculty of Science, Technology and Arts

Department of Computing

Project (Technical Computing)

[55-604708]

2019/20

| | |
|--------------------------|--|
| Author: | Jonathan Cunnew |
| Student ID: | 26027956 |
| Year Submitted: | 2020 |
| Supervisor: | Michael Bass |
| Second Marker: | Soumya Basu |
| Degree Course: | Computer Science for Games |
| Title of Project: | Creating an easy to use work tracker for staff and students |

Confidentiality Required?

NO ☒

YES ☐

Acknowledgments

I would like to express my gratitude to my project supervisor Michael Bass. For the entirety of the project he has helped me develop my writing skills and have pushed me to work as hard as I could to develop a successful project that I am proud of.

Also, I would like to thank my friends and family for helping me along the journey through the four years of university. This project would also not have been possible without their support.

Finally, a special thank you to my partner. They have pushed me to keep working even when times were tough and for putting up with the long nights of glowing screens keeping her awake at night.

Abstract Synopsis

The idea of this project is to help bridge a gap between teacher and student around the subject of homework. The scope of the project is to have a working, prototype, Android application that both teachers and their students can use to help manage homework tasks. The main aim is to make a very user-friendly application that students of many ages can use easily and efficiently. To do so the application will aim to include notifications and simple design layouts that so many people can easily use the application. The standout feature that the app will offer is the “homework creation” section on the teacher end. The teachers will have access to be able to make and set tasks that will be sent and listed easily for their linked students. The students will then be able to see these tasks and have all the information available to them whenever they want. As the students receives homework tasks, they can respond to such task by asking for help, feedback or to just complete the assigned task. By gaining access to notifications the students can get feedback quickly and reliably helping them to complete homework.

The Application will be made using Android studio and the programming language Java. These were chosen due to their relevance within the industry and the wide range of available learning resources. The prototype aims will be to have a database driven login system using php as the language of choice. Along with a task management system that is assessable for both teachers and students.

Contents page

Table of Contents

| | |
|--|----|
| Acknowledgments | 2 |
| Abstract Synopsis..... | 3 |
| Table of Figures | 6 |
| 1 Introduction | 8 |
| 1.1 Background..... | 8 |
| 1.2 Aims & Objectives | 8 |
| 1.3 Planning Tools | 9 |
| 1.3.1 Version Control..... | 9 |
| 1.3.2 Project Management Software..... | 9 |
| 1.3.3 Project Schedule | 10 |
| 2 Research..... | 10 |
| 2.1 Existing Management Applications | 10 |
| 2.1.1 Example One – Show My Homework | 10 |
| 2.1.2 Example Two – Blackboard..... | 11 |
| 2.2 Need for Such a Program | 12 |
| 2.2.1 Benefits of An Online Homework Tracker | 12 |
| 2.2.2 Reliance on the Internet & Online Systems..... | 12 |
| 2.2.3 Updating Schools to online recourses | 13 |
| 3 Development Tools | 13 |
| 3.1 Android vs IOS | 13 |
| 3.1.1 Benefits of Android..... | 14 |
| 3.2 Mobile App vs Web App..... | 14 |
| 3.2.1 Native Mobile Features | 15 |
| 3.3 Android Studio..... | 15 |
| 3.4 Programming Language..... | 15 |
| 3.4.1 Android Language..... | 15 |
| 3.4.2 Database Language..... | 16 |
| 4 App Design | 16 |
| 4.1 Device Compatibility..... | 16 |
| 4.2 Accessibility | 17 |
| 4.2.1 Universal Design Principal | 17 |
| 4.3 Storyboards | 18 |
| 4.4 User Interface..... | 18 |
| 4.4.1 Ease of Use..... | 18 |
| 4.4.2 Material Design..... | 18 |
| 4.4.3 Existing Designs | 19 |
| 4.5 Navigation | 20 |
| 4.5.1 Teacher and Student..... | 20 |
| 4.5.2 Navigation Bar | 21 |

| | |
|---|----|
| 4.6 Database Connectivity..... | 21 |
| 4.6.1 phpMyAdmin & SQL | 21 |
| 4.6.2 In App Connectivity..... | 21 |
| 5 Development..... | 21 |
| 5.1 Setting up the Development Environment | 21 |
| 5.1.1 File Structure..... | 22 |
| 5.1.2 Android Emulation..... | 22 |
| 5.2 Learning the Basics of Android Studio..... | 23 |
| 5.3 Layouts | 24 |
| 5.4 Functionality..... | 24 |
| 5.4.1 Basic Activity's..... | 24 |
| 5.4.2 Bottom Navigation & Fragments | 26 |
| 5.4.3 Teacher and Student Views | 27 |
| 5.4.5 Database Connectivity Using php..... | 30 |
| 5.4.6 Account Creation and Login..... | 31 |
| 5.4.7 Task Creation & Task Viewing..... | 37 |
| 5.4.8 - Update and Delete..... | 41 |
| 5.5 Database..... | 42 |
| 5.5.1 Local and University..... | 42 |
| 5.5.2 Xampp set up | 43 |
| 5.5.3 Users | 43 |
| 5.5.4 Tasks | 44 |
| 5.6 Debugging..... | 45 |
| 5.6.1 Application Problems..... | 45 |
| 5.6.2 Database Problems | 46 |
| 6 Testing..... | 47 |
| 6.1 User Testing Environment | 47 |
| 6.2 User Feedback | 48 |
| 6.2.1 Suggested Changes | 48 |
| 7. Critical Reflection | 48 |
| 7.1 Project Planning..... | 48 |
| 7.2 Future Development | 49 |
| 7.3 Conclusion | 49 |
| 8. References | 50 |
| Appendix A - Project Specification | 53 |
| Appendix B – Ethics Form..... | 56 |
| Appendix C – Abbreviations and Definitions/Glossary | 61 |
| Appendix D – Physical Application Designs..... | 62 |
| Appendix E – larger screenshots | 66 |

Table of Figures

| | |
|---|----|
| Figure 1 - Reviews of Show my Homework from the Google Play store | 11 |
| Figure 2 - Review showing praise of parent interaction | 11 |
| Figure 3 - Review of Tapestry from the Google Play store | 12 |
| Figure 4 - Reviewer unable to use online tracking due to high traffic demands | 13 |
| Figure 5 - Snapshot from Android studios 'configure your project' page | 17 |
| Figure 6 - A section of the Material Design components list | 19 |
| Figure 7 - Example of YouTube's bottom navigation bar | 19 |
| Figure 8 - Example of Blackboards Navigation bar | 20 |
| Figure 9 - example of the emulation running the project application | 22 |
| Figure 10 - Project examples page | 23 |
| Figure 11 - Code example, linking to login layout | 23 |
| Figure 12 - Example image of an attribute being constraint | 24 |
| Figure 13 - Code and layout for the first page of the application | 25 |
| Figure 14 - Example of a listener function used throughout the project | 25 |
| Figure 15 - Example of a function used to move to a new activity | 26 |
| Figure 16 - Image of teacher's manager class code | 26 |
| Figure 17 - Teachers navigation bar | 27 |
| Figure 18 - Example of additional steps to give a layout | 27 |
| Figure 19 - Fragment to new activity | 27 |
| Figure 20 - Additional fragment hierarchy example | 27 |
| Figure 21 - Original first page & account creation page | 28 |
| Figure 22 - Radio button grouping | 28 |
| Figure 23 - Radio button usage when creating an account | 28 |
| Figure 24- Comparison of Teacher and Student navigation bars | 29 |
| Figure 25 - Images of Create Task & Account pages | 29 |
| Figure 26 - Localhost link | 30 |
| Figure 27 - Php constants | 30 |
| Figure 28 - Php connect | 30 |
| Figure 29 - Android Php constants | 31 |
| Figure 30 - RegisterUser.php - data checking | 32 |
| Figure 31 - RegisterUser.php - pushing data to query | 32 |
| Figure 32 - Creating a user INSERT query | 33 |
| Figure 33 - Login POST | 33 |
| Figure 34 - SELECT query checking user login data | 33 |
| Figure 35- Example of SELECT* and fetch_assoc() in user login | 33 |
| Figure 36 - Extract of applying fetched data to a JSON object | 34 |
| Figure 37 - Reading text inputted by the user | 34 |
| Figure 38 - HashMap for Registering a user | 35 |
| Figure 39 - Example of JSON object message | 35 |
| Figure 40 - Login HashMap | 35 |
| Figure 41- User account manager class | 36 |
| Figure 42- Logging in JSON response | 36 |
| Figure 43 - Application checking if the users is a teacher | 37 |
| Figure 44 - Log out function | 37 |
| Figure 45 - INSERT query for task creation | 37 |

| | |
|---|----|
| Figure 46 - Inputting the task owner | 37 |
| Figure 47 - RecyclerView of Task cards | 38 |
| Figure 48 - Task collecting array | 38 |
| Figure 49 - JSONArray for collecting all tasks | 39 |
| Figure 50 - Checking for the owner of a task | 39 |
| Figure 51 - Passing data via Intent to expanded task page | 40 |
| Figure 52 - Teacher and student views of a full task | 40 |
| Figure 53 - Updating a task php | 41 |
| Figure 54 - UpdatePass php function..... | 41 |
| Figure 55 - Updating password pages..... | 42 |
| Figure 56 - DeleteTask function in php | 42 |
| Figure 57 - Image of java security error on app | 43 |
| Figure 58 - Image of file structure for Xampp..... | 43 |
| Figure 59 - Image of user's table from the database..... | 44 |
| Figure 60 - Image of tasks table from the database | 45 |
| Figure 61 - Fragment getActivity() | 45 |
| Figure 62 - Calendar window | 45 |
| Figure 63 - Navigation bar bug..... | 46 |
| Figure 64 - Problematic while loop | 47 |

1 Introduction

1.1 Background

The idea behind an easy to use work tracker came from volunteering at a small school in Leeds that aims to teach coding to a range of age groups all with different learning abilities.

While volunteering, the owners wanted to test out Code Avengers with one of the coding classes, since most of the students were getting close to finishing the Scratch material that they had planned out. Code Avengers featured five to ten-minute lessons that were easy to digest for the students. Most of them enjoyed the structure and the feedback of seeing what they had completed and what was still to be done.

Together with the owners and some other volunteers, we talked about what we liked and disliked while using Code Avengers with the students. We all agreed that the lesson structure and the progress tracking would be helpful when trying to keep up with range of learning differences between the students. An open discussion was created amongst the group about whether we could make a tracking system for multiple different applications or websites. The discussion also involved the work tracker being easily accessible for all types of abilities and ages of the students. This would ensure everyone was able to navigate the system and have no problem using it.

A browser extension which linked to the current webpage was the initial concept. It would track the position of a webpage so that when a student is following an online guide or tutorial, both the student and teachers are able to view the progression of the online resources. This idea was suggested since at Geeks Room CIC, most lessons involved the students accessing a desktop computer to learn to code and participate in other activities, such as group gaming.

When starting to plan, the initial idea of a browser extension concept changed to be more focused on creating a mobile application. This design change would allow the students to have more flexibility in managing their work along with allowing parents to interact more with their child's work.

1.2 Aims & Objectives

Aims:

The aim of the project is to create a mobile application prototype that will run on Android devices. The Prototype should include a user login system for both teachers and students with each one giving different features of the application. Along with a login system the teachers shall be able to create tasks and connect their students to such tasks. Students will be able to receive the created tasks once they login to the application and check off if they have completed such task.

Additionally, the application will need to be clear and easy to use for everyone. I will aim to do this by having a simple user interface design along with a recognisable design layout that most people should find familiar. An extension to the initial aim is to have access to mobile permissions such as camera access and notifications.

Objectives:

- Be able to create a database driven login system that gives different access for teachers and students

- Assign teachers the ability to create tasks that will show in their student's task lists
- Add mobile permission to the app, such as camera and notifications to make the experience of the app better
- Design a user interface that follow conventional application styles that is easy to use
- Gain feedback about the user interface and change where applicable

1.3 Planning Tools

1.3.1 Version Control

Version control allows a user to view changes made to files and recall to specific versions of such files if needed to. The use of version control will be useful for my work on this project as it will allow me to keep a record of changes I have made and revert if needed.

The version of source control being used will be Git. Git is one of the most popular versions of source control systems there is and is easily accessible to everyone for free (RhodeCode, Inc, 2016). Via the university, an enterprise account with Github was created, giving access to unlimited private repositories. Github is a hosting company for people wanting to use Git without the need to learn commands inputs via their pc's console.

Github can be accessed by their webpage or through the Github Desktop application. the application was chosen to manage the commit history and push all the changes to any code. The application made managing the repository much easier than the web client and is the reason for choosing to use it over just the web client. A Gitignore file was created to make transferring between workstations easier. This file tells the repository to not upload any files that are not needed to make the project run. Including this means smaller file upload and download when swapping workstations. Commits will be done regularly to avoid any loss of work and will be accompanied by a title and description of what was achieved with the section of code.

1.3.2 Project Management Software

When starting to plan a project, handwritten notes help to speed up the process of designing and planning the initial structure of the project. Application designs were drawn out to help visualise how the structure of the application would look like. Additionally, the main tasks for developing the project were written down and then compiled together into a Trello board. This meant all the notes were accessible at all time.

Trello is an online project management tool that lets the user's manage small task by organising them into categories and giving them due dates. Trello was chosen to manage the project because there was previous experience of using the software and found it helpful when working as a group. Trello is also a very popular tool to use in industry so getting so further experience with the software is always welcome (Clyde, 2019).

Scrum was used for a few different projects within university as the project management methodology of choice. The meetings at the start of session were enjoyable and helped to set the goals for the day. However, trying to use Scrum within a single person project didn't work as intended. Sticking to the tasks set at the start of the day became troublesome as development would always steer toward the next quickest thing to develop. Swapped to an Agile approach of

developing the application help to keep track of all the tasks that needed to be done. This allowed for consistent testing of the application as well, helping to build a working application day after day.

1.3.3 Project Schedule

Supervisor meeting were set up every two weeks to discuss the progress made with the project as a whole. In these meeting notes were taken about the quality of the in the past two weeks and also about what needed to be done for the next meeting. For each meeting questions and a working version of the application will be prepared to save time, so that as much feedback can be received.

A time plan can be seen in Appendix A – project specification.

2 Research

2.1 Existing Management Applications

Before undertaking the project, research into already existing applications/ website needed to be done. This would provide a better understanding of what would need to be accomplished and what standards users are looking for with in their management applications.

2.1.1 Example One – Show My Homework

Show my Homework is a work tracking application and website that provides similar features to what the project intends to create (Satchel, inc., 2020). The ability to produce homework tasks for students to be able to view and work on via a mobile device or computer. Due to the similar functions to the project application the comparison to an already existing services will provide a good insight to the quality that need to work towards. Seeing that there is already an application that provides a similar experience to what the project intends to develop, the need for such a software is present with in the teaching industry as the company running Show my Homework states the program is used in over 1600 schools (Satchel, inc., 2020).

Reviews from the Google Play store shows that some users find the application for tracking homework's helpful and a great impact on schoolwork.

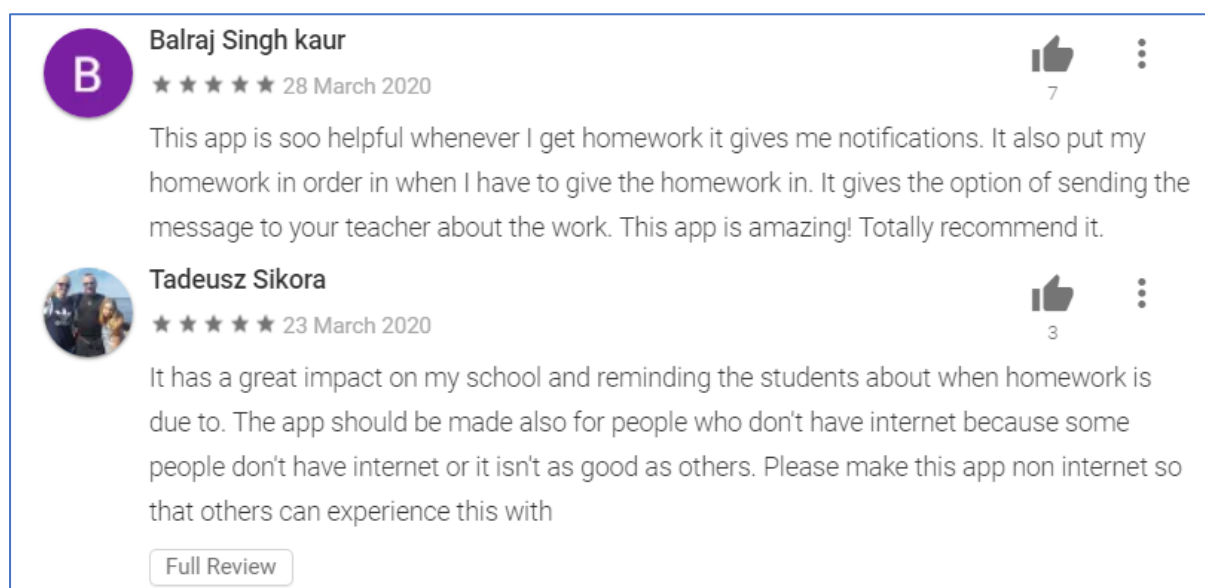


Figure 1 - Reviews of Show my Homework from the Google Play store

Show my Homework is aimed towards secondary schools to be able to provide the teacher student and parent interaction (Satchel, inc., 2020). While aiming to a smaller scale than university, the software can provide the features to the parents giving a much more personal approach to keeping track of a students work. While working on a small scale, this is something parents are liking with Show my Homework and are expressing their thoughts about.

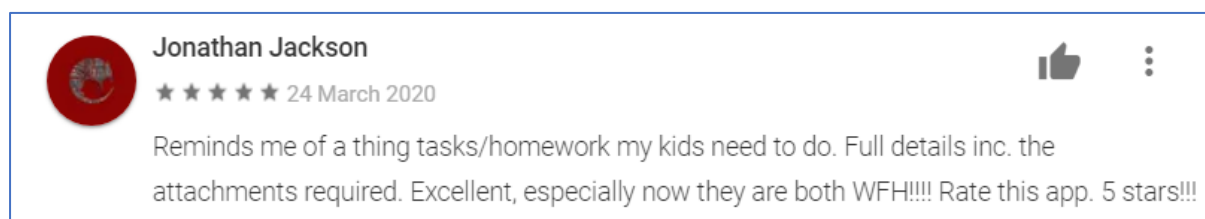


Figure 2 - Review showing praise of parent interaction

2.1.2 Example Two – Blackboard

Blackboard is another learning management system that allows teacher and students to interact together via the sharing of work and announcements. Blackboard user base as of 2017 is reaching over 100 million users, (Blackboard, inc., 2017) making it one of the largest online learning management software available. Blackboard is mainly a web-based application with a partner application for mobile devices. However, the mobile application is limited to viewing work and announcements.

Blackboard offers a much larger scale of task management than other examples and what this project intends to do. Seeing a difference in scale compared to Show my Homework shows that the need to manage and track students work is needed within small groups of teachers and students, along with larger organizations such as universities.

2.2 Need for Such a Program

2.2.1 Benefits of An Online Homework Tracker

Studies have shown that there are some obvious benefits to using Virtual Learning Environment (VLE). Some of these benefits include time saving for marking, a reduction in cheating and less work being left incomplete (Trussell, 2020). The ability to save time with online work checking gives the teachers longer to arrange and set new tasks for their students along with additional time to respond to questions asked by their students. The benefits of using a VLE extend past the use of teachers and students, reaching out to the parents as well. As shown previously, Show my Homework provides access to work tracking for parents giving them the ability to see and be notified of tasks set for their children. Tapestry, another work tracker for younger children, also provides parent interactions and giving them control over what their children are learning.

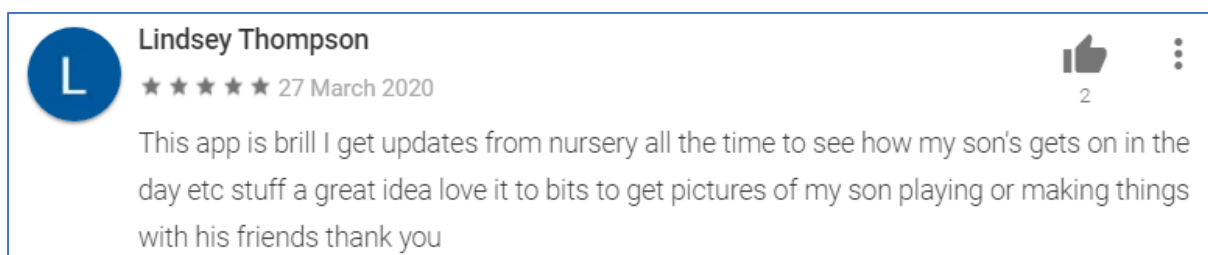


Figure 3 - Review of Tapestry from the Google Play store

- This review is supporting the want for parent interactions within work tracking applications

Giving the teachers more time to teach and revise work is a key feature of online work trackers. By providing a simple and easy environment for teachers and student to look up and submit work creates a faster working for teachers to spend more time working in person with their students. This is supported again from Firefly, another company providing a VLE to aid teachers and students with work tasks, by stating an increase in teaching time and amount of hand in of work (firefly, inc., 2020).

By seeing the increase of productivity in teachers and students, the transition to an online system for work tracking in schools is outweighing the need for paper homework. An addition to using an online system for work tracking means that all students always have access to work. Missing lessons or being ill is a cause of missing homework while using paper, but while using an online system gives access to all at any time (Jonsdottir, 2017).

2.2.2 Reliance on the Internet & Online Systems

While the move to online systems is showing an increase in productivity, the need to have access to the internet always is a risk factor that needs to be considered. Only 93% of the UK always has access to the internet within the household (Statista, inc., 2020). Without access to the internet the ability to see any tasks set is impossible. This potentially can lead to some students missing out of seeing tasks being posted causing them to miss out on the work entirely while at home.

A second risk of the reliance of the internet is also relying on the software being used can handle as many users as possible. A strong example of this is with Show my Homework and the 2020 COVID-19 virus outbreak. Due to the UK lockdown taken effect in March, Show my Homework usage has risen

drastically, cause users to be unable to use the application simply because of the number of online users at one time.

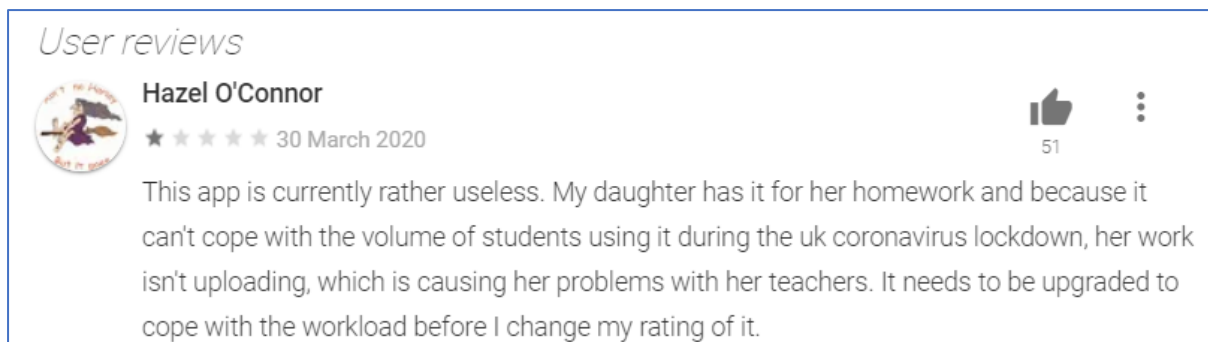


Figure 4 - Reviewer unable to use online tracking due to high traffic demands

- This review, along with others, show that an unexpected increase in a user base can cause issues that affect many more people unintentionally.

2.2.3 Updating Schools to online recourses

Studies and user reviews have shown that using a VLE can save time for teachers and students. However, the cost and time investment to develop the schools to a substantial point to achieve this productivity is a costly one. The UK government is said to be increasing the funding for all schools in 2020, enabling all schools to spend the money upgrading and improving all aspects of their teaching (Department for Education, 2019). Even with the increased funding, studies show that 28% of secondary schools with it the UK are in financial trouble (Weale, 2019). With additional funding but schools still being in financial trouble, the cost to upgrade to a VLE is not an option for everyone.

The cost of virtual learning environments is never fully disclosed to the public. However, users of Blackboard have commented on the price being around \$160,00 per year for the full Blackboard package. With a high price tag schools may not be able to afford the upgrade, leaving paper homework and management the only option.

With all the virtual learning environments training is needed to be able to make full use of the software. An Assistant Principal reviewed Blackboard saying the software was easy to navigate and appeals to students but needed training to master the features and was not friendly for people not very technologically savvy (DiBiasio, 2020). Teachers and students needing to learn new software is the key to increasing the productivity while using them, and if these software's take too long to learn, then can they efficiently be integrated into schools easily.

3 Development Tools

3.1 Android vs IOS

When choosing between developing on Android or Apple iOS devices, consideration was needed for how and if access to the development tools were possible. I personally use an iOS mobile device for my day to day use and have always been interested in the steps need to start development. However, after some research into iOS development, the programming language used was SWIFT. A language only available on Apple computers (Apple, Inc, 2016). This posed some questions, install Apple's MacOS on to my Windows 10 pc or use the unfamiliar Android operating system?

To start development of an Android app there was a single download for Android Studio. This is a development software that can also emulate Android devices on your pc (Google, Inc., n.d.). The ease of a single download was much more appealing, as it meant I could start learning the software sooner without the need to install MacOS on my home pc

An advantage to developing on iOS is Apple have developed a version of iOS that can be run on most iPhones. This means more phones can be accommodated by the same version of iOS, providing a much easier choice when deciding on device compatibility of the project (StatCounter, inc., 2020). On the android side of this, the spread of versions is much higher, with a lot of devices currently running a lot of different Android versions. This setting up the project to be able to accommodate as many users as possible would be a more difficult task (StatCounter, inc., 2020).

3.1.1 Benefits of Android

When developing for Android devices there are two programming languages that can be used, Java and Kotlin. Java is an old and well known programming language that has been adapted well into the development of Android applications (Langly, 2002). This choice of programming language is a benefit over developing for iOS since being given the option to use a well-known language over the single Apple specified language, opens up the ability of finding many more guides and tutorials online to help learn and develop the project.

Another benefit of developing on Android is the cost to develop and publish apps. There is a one-time fee of £25 to publish an app on the Google play store whereas to develop and publish an app on the App store there is a fee of \$99 per year (Mackenzie, 2012). Although the fee is higher to publish on the app store, the percentage of revenue that Apple takes from sales is smaller than Google. Also, the popularity of the app store over the play store has shown greater results in sales, helping to balance out the higher fee (Komal, 2019).

3.2 Mobile App vs Web App

Another question that needed to be asked was to develop for a mobile device, such as a smart phone or tablet or develop a web-based service. This question came about because the original concept was to develop a website or browser extension. By developing a web app, services that may already be using in schools would be much easier to incorporate into the development of the application. Another benefit to developing a web app would be the theoretical accessibility to reach almost everyone with access to the internet. This would be good as it would mean anyone able to connect to the internet could access the application.

When considering how to develop the mobile application, consideration of the devices owned by the general public was taken. The percentage of people owning smart phones has now risen higher than desktop computers (StatCounter, inc., 2020). This shows that trying to reach more people by developing a web app is not as practical as first thought.

The decision was made against developing a web app as the features a smart phone / tablet can offer are far greater than what could be accomplished with a web app. The reachability and mobility of a mobile device outweighs what could be done with a web app.

3.2.1 Native Mobile Features

The native features that a mobile app can offer would be greatly beneficial to the users. The features would open the possibility of using the devices camera and push notification services. By gaining the permissions to access the camera and push notifications, the users would be able to upload photos of work to prove that the tasks have been completed. Along with being notified as soon as possible when new tasks have arrived.

Being able to upload photos to the project app would mean teachers could take pictures of already existing work and upload it to the app. This opens the possibilities of what sort of tasks can be sent via the app and what responses can be accepted. Push notifications give the user instant access to seeing new tasks posted by a teach. Meaning the students would not need to be told in person about upcoming and new tasks (Summerfield, n.d.).

3.3 Android Studio

Android Studio is the official integrated development environment (IDE) for developing Android applications on Googles Android operating systems (Google, Inc., n.d.). Android Studio offers a visual layout editor and emulation of android devices. These two features were the deciding factor when choosing to use the program. The layout editor is a fully featured visual representation of designing the look of applications. This was beneficial as layouts in Android studio are made using Extensible Markup Language (XML) programming language. XML is a programming language that I had seen before when learning website design in university but never expanded much on it. Having the ability to use a visual editor to help make the XML files took some of the pressure off learning my chosen language when developing the application.

Additionally, by not owning any Android devices, the built-in emulation of Android Studio would be greatly beneficial and easily provides the opportunity to test the application consistently without the need to by a Android device.

Some other development platforms for making Android apps included a free to download the software. Android studio is free to use. This is another reason that it was my choice of development software (Slank, inc., 2020). Other programs such as Visual Studio offer the ability to design and make an Android app but lack the features to help me learn the new programming languages necessary to make the application.

3.4 Programming Language

Both the application and the database used to connect to the application, have a choice of different languages to develop with. Deciding between that languages was a key choice as changing language in the middle of development is not an easy task for anyone. Picking a language that could be learnt quickly was top priority.

3.4.1 Android Language

When choosing the language to develop with inside of Android Studio, there are two options, Java and Kotlin. Java is a well-known general-purpose programming language used by over 12 million developers worldwide (Oracle, inc., 2020). Java is a programming language that has been used worldwide for many years, meaning there are guides and books to learn the language all over.

Kotlin is a new programming language developed by JetBrains (JetBrains, inc., 2020). Kotlin is an accepted language by Google when developing Android applications and is recommended for people brand new to app development. Kotlin boasts the ability to make fast applications as it is a language built from the ground up for the purpose of running mobile applications (Google, inc., 2020).

Java was chosen for the development language because of the vast number of guides available. Because Kotlin is still a recent language, the number of guides that can be found are largely limited to the official documentation. Whereas with Java additional guides from YouTube and all over the internet can be access easily. However, due to the high quantity of guides, finding a reliable good practice Java guide would be difficult.

3.4.2 Database Language

After choosing a language to use for Android Studio, the choice of language needed to be decided for the database. The software chosen to host the database was phpMyAdmin. phpMyAdmin is a free software that handles MySQL requests over the web. phpMyAdmin Is a lightweight software designed to simplify hosting a database (Bennetch, 20202). The university provided every student with a phpMyAdmin database on the university servers. This was the standout factor when researching about databases as I had very little understanding about databases when starting this project.

The alternative considered was using MySQL directly. Using MySQL directly would mean having access to a lot more feature than using a lightweight software (Oracle, inc., 2020). However, this was a little overwhelming since learning Java and Android Studio was already a large task. Therefore, the decision was made to use phpMyAdmin. phpMyAdmin is written in the web programming language php and uses php requests to send and receive data from the database. Php is also a language that was shown to inside of the university, so a basic understanding of the language was already there. This again helped to be able to spend more time learning Java and Android Studio.

4 App Design

4.1 Device Compatibility

Being able to reach as many people as possible was one of my top considerations when designing the application. The application is simple in features and will not require a lot of the new hardware found in newer smart phones and tablets. When setting up a new project in Android Studio, you are asked to pick a minimum application programming interface level (API). Because the application would be lacking in new features to push hardware, a lower level of API could be chosen. This would make the application more widely accessible as it would run an more devices owned by the public. Initially the plan was to use Android version 4.3 (Jellybean), as the official Android documentation stated that this version would cover 98.4% of devices (Google, inc., 2020). At first glance being able to reach this many device would be great. Covering this many devise would ensure that as many people as possible would be able to access the application with little to no trouble. However, upon further investigation, using this API would mean missing out on some key features that are commonplace in today's culture. These features included transition animations and push notifications to locked phones.

Transition animations are a staple of app design and enhance the user experience, they help to provide feedback to the user that their actions have taken place (Babich, 2018).by using a API level that does not allow animations, the users experience of the application would suffer. Potentially causing frustration with the application. Also, having the ability to push notifications to the user while their device is not currently active was a must. If students could be notified immediately upon receiving new tasks or changes to an existing task, the student's management experience would be greatly enhanced.

To be able to use these additional features, the minimum API level needed to be raised to 5.0 (Lollipop). This however would mean losing some device coverage. Bringing the approximate percentage to 85% of devices able to use my application. As seen in Figure 5 below. API level 5.0 was the final choice because this was the first version that allowed the use of push notifications (Google, inc., 2020). This API still gave a large coverage of devices and offered the features that are intended to use in the application. According to StatCounter.com, the percentage of people using versions of android lower than 5.0 is less than 5% (StatCounter, inc., 2020)

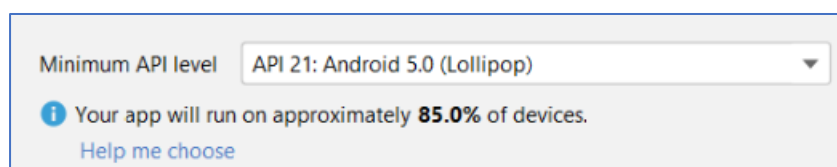


Figure 5 - Snapshot from Android studios 'configure your project' page

4.2 Accessibility

Because the application is aiming towards a lot of different age groups, the need to make it as accessible as possible was important. Implementing small changes such as increased font size and large text boxes for typing information, helps the users read and navigate the application easier. Additionally allowing newer devices to use the inbuild features, such as screen readers and magnifiers is another step towards getting as many people as possible able to use the app (Willson, 2018). To design the application, the Universal Design Principles will be taken into account to ensure the application is assessible and easy to use foe everyone.

4.2.1 Universal Design Principal

The Universal Design Principles is a set of guidelines to help when designing products or environments to make sure that everyone is accommodated properly before development has started. By following these guidelines, creating an application that is easy to use for a wide range of people would be easier as a guide of accessible options would already be laid out. (Center of Universal Design in North Carolina, n.d.).

Increasing font size and margins of error of input boxes helps with meeting the guideline, 'Flexibility of use'. This helps to facilitate the user's accuracy and precision. By following just one of these guidelines is a large jump in accommodating as many users as possible. Additionally, by following a single colour code and minimising the number of colours used in the development can greatly aid uses with learning disabilities

4.3 Storyboards

To help design the look of the application, designs of the main pages were made using pen and paper as the ease of drawing physically was much easier to portray the design chose that were thought of. The 4 main pages for the teachers view and suggestions for the initial log in pages were drawn out. These designs plans helped to follow the Universal Design Principles of accommodating lots of users. When designing the page layouts, inspiration was taking from looking at existing management applications and adapting what was good and beneficial for the application design.

Scans of the paper designs can be found in Appendix D.

4.4 User Interface

4.4.1 Ease of Use

Developing an easy to understand user interface was one the key points of development. With approximately 15% of the world's population having some sort of disability, the need to make my app easy to use was to help avoid any confusion on the users end when using the app in a learning environment (Geerlings, 2019).

By following the Universal Design Principals and the Material Design guidelines (mentioned in next section), simple yet effective changes such as text size and colour design will make the application easier to use in the long term.

4.4.2 Material Design

Material Design is a website created by Google to aid developers follow industry guidelines when developing applications. These guidelines have been created using the “best practices” of user interface design. Material design offers an extensive list of design guidelines for all aspects of app design. The website covers all areas that will be included into the application. Material design was chosen to follow for a design principal because the site is developed by Google, with collaborations of many different developers to help provide the best possible looking apps, (Google, inc., 2020). When trying to find an alternative to Material design, there were results from many different companies showing what they felt was good app design but didn't provide such an extensive list of design concepts (Blair, 2020). However, Material design will not be relied upon and common standards from the applications mentioned in the next section would be used to design the look of the application.

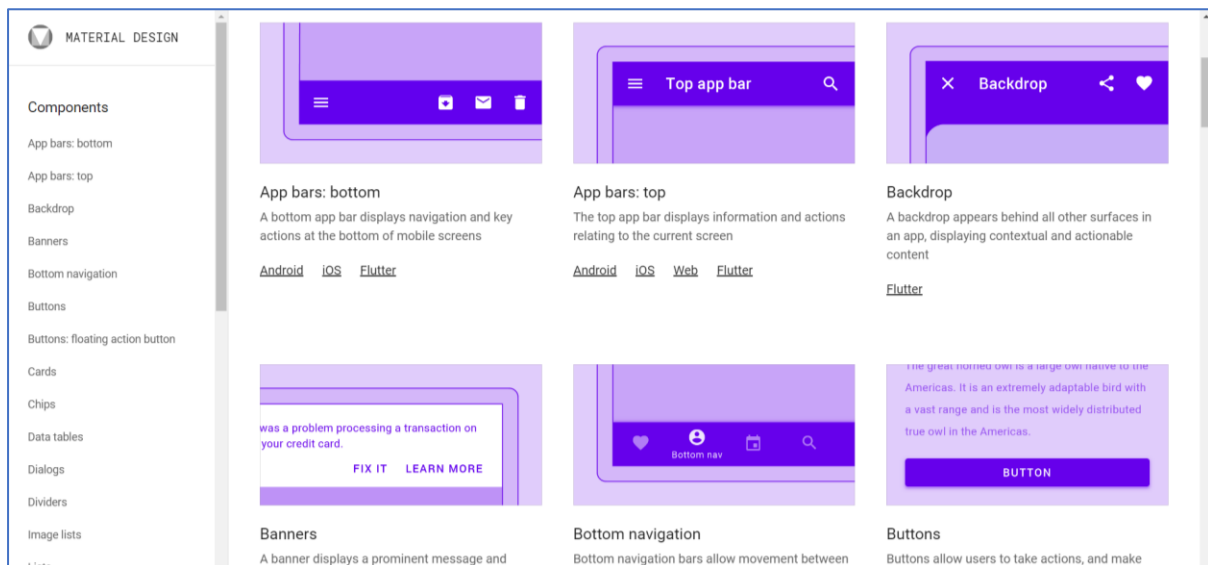


Figure 6 - A section of the Material Design components list

4.4.3 Existing Designs

When designing the look of the application, existing applications were investigated that are commonly used by the general public and some that are more specific to project management. Some applications such as Facebook, Twitter and YouTube all use bottom navigations bars to help the user navigate the applications most important pages. Finding multiple apps using bottom navigation bars as a means of navigation was proof that the feature was common practice enough to use in the application. These three apps also followed the Material Design guidelines as talked about previously, reinforcing why following the guidelines would be beneficial to my application design (Apple, Inc., 2020).

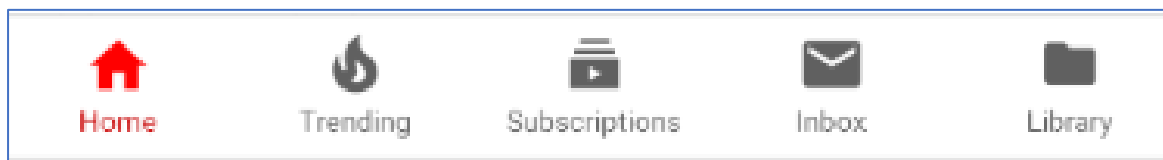


Figure 7 - Example of YouTube's bottom navigation bar

- the 5 key pages laid out and easy to reach

Applications that feature work tracking or planning were investigated next. Microsoft OneNote was an application investigated into. The application again uses principles from the Material Design guideline and has received great reviews regarding the design of the application on the Apple app store. (Microsoft OneNote Ratings and Reviews, 2020).

After looking at designs of applications that were familiar, investigations into the designs of other project management apps started. The design of blackboard was investigated. By looking into just the design of the application, noticing the easy to use features was simple. The navigation bar of Blackboard was hidden behind a menu located in the top left, this meant that reaching the menu was an annoyance when using larger devices one handed. Seeing this helped to reinforce that I wanted to design my application with a bottom navigation bar in mind.

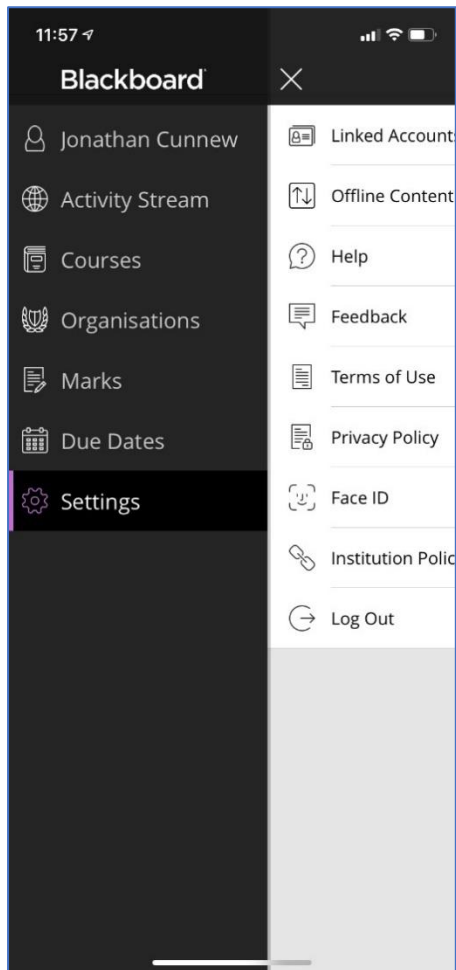


Figure 8 - Example of Blackboards Navigation bar
 – Hidden behind a button situated at the top left of the page

4.5 Navigation

4.5.1 Teacher and Student

To avoid overcomplicating the application, the interface was designed to be different for teachers and students. By doing this the teachers view of the application could be more complex to accommodate the ability to create tasks and users. Removing these features from the view of the students means a less complicated user interface for the students. The ability to create tasks and students is not a feature that should be given to a student account anyway.

Separating the teachers and students was an early design choice and a lot of the ideas behind the application came from this choice. A negative impact in separating teachers and students was the need to design some of the pages twice. However, the additional work to make additional application pages also meant the design the teacher and student views could be completely differently if needed.

4.5.2 Navigation Bar

Navigation bars are extremely useful when designing an application that focuses on a few standout pages (Google, inc., 2020). Since the application was designed around using four main pages: Task list, create task, create account and user account pages, quickly being able to swap between only 4 different pages was a must. This was done with a navigation bar. The navigation bar allows the four important pages of the application to be all together at the bottom of the app, where they are easy to reach and easily distinguishable. Being able to see and reach all 4 of the most important pages of the application means users will be able to learn the application quickly and easily (Apple, inc., 2020).

4.6 Database Connectivity

4.6.1 phpMyAdmin & SQL

PhpMyAdmin uses and manages SQL queries to help the user send and receive data from a database. The intention was to use the university provided phpMyAdmin server to host the database, as it shows the ability to connect to a live server over a locally hosted one. If the university server is unable to connect to, then additional 3rd party software can be used to create a local server and database. A local server was set up using Xampp anyway, in case the university server had issues. This was useful because the local server would be a reliable back up to use. Xampp is an open source software developed to test databases and websites before being uploaded to a remote web server (Apache Friends, inc. , 2020). Because phpMyAdmin can be used easily with Xampp, it meant set up of a similar testing area to the university would be possible.

4.6.2 In App Connectivity

The connection between the application and database was simple in concept. The application would connect to the database and be able to store users' names, emails, password and if they are a teacher or student. With this information stored, a login system could be created using the email addresses created in the app (real email addresses would not be needed or created inside of the application) and passwords, encrypted with MD5. MD5 is an encryption method that easily accessed with in php development and is a fast and highly efficient encryption method. A second database table was needed to manage the user task lists. Each task would hold information such as a title, description, due date and owner. Connections between the android application and the database will all be handled within php files that I will create outside of Android Studio.

5 Development

Throughout development, the official documentation for Android Studio was used to aid the development of the project (Google, Inc., n.d.). Additionally, the official documentation for php was used for writing the php code (the PHP Group, 2020).

5.1 Setting up the Development Environment

When starting development, a plan was needed to figure out where I was going to develop the application and how it would be organised. Github allowed the transfer of project files easily, meaning transfer workstations was not a problem. The Github repository was a great way to back up

any progress but additional copies would be on an external hard drive to avoid any work loss. The local server was set up the same for both workstations at the same time to make sure server setting were not a problem when working on the database.

5.1.1 File Structure

Planning the file structure of the projects was something done before starting work in Android Studio. Doing this helped to visualize the structure and the scope of the project. As many Java classes were made to mate up with the designs drawn beforehand with additional XML layout files for each.

5.1.2 Android Emulation

For developers that do not own a physical Android device, Android Studio includes a built-in emulation software. Emulation is one system is imitating or reproducing another system. By emulating 2 different Android devices, a 5.2-inch screen and another with a bigger 6-inch screen, the project could be tested on a small but different range of screen devices. Android Studio device emulation included lots of customisable setting to be able to set up the preferred preferences to match the planned target API, as described before. Android Emulation gives full control as if it was a physical device, meaning testing the application did not suffer due to the lack of a physical device.

However, running the emulator was sometimes a slow process. Occasionally the emulation would fail to load on one of the computers being used and would need be reset in order to work again. Wiping the data off an emulation is quick but the reinstall took extra time that was an annoyance. This seemed to be the only fix for the problem.

Having access to a physical device for testing would help to reinforce the stability of the application on a real device. Along with seeing the stability of the application, a physical device could test if the design of the pages were correctly displaying running off screen.

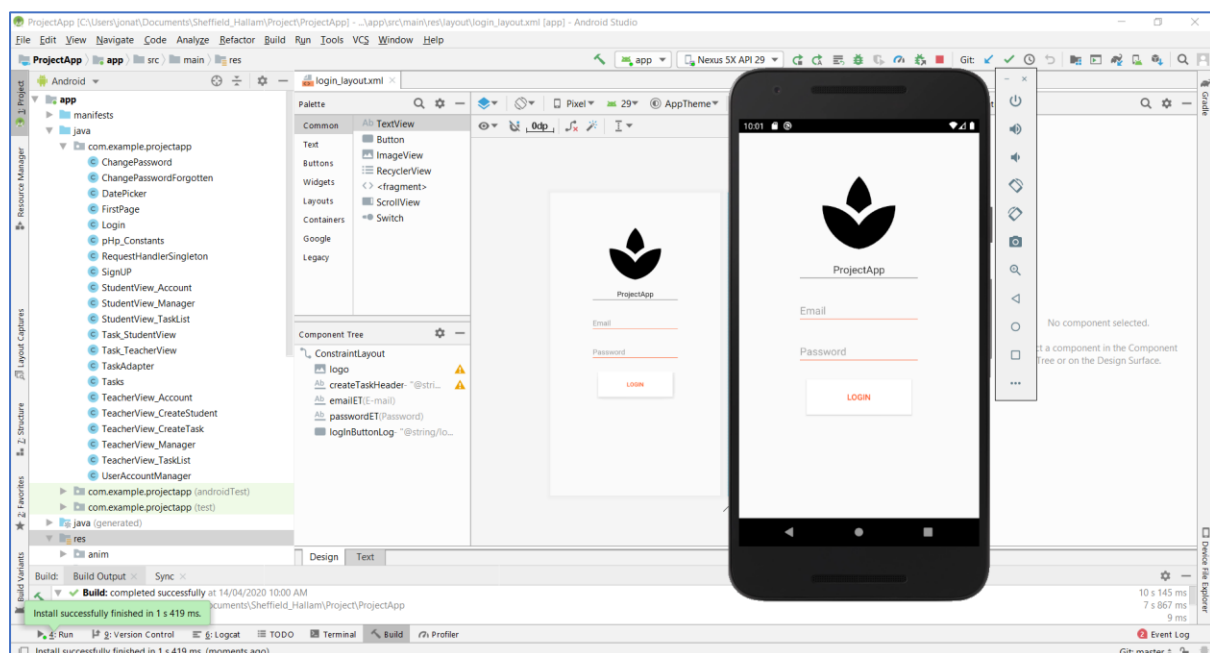


Figure 9 - example of the emulation running the project application

5.2 Learning the Basics of Android Studio

Starting the development of the application was going to be a large task. The basics of Android studio needed to be learnt before the development could start. The software offers a set of prebuild projects that developers can start work with. By starting the project from a template, some code could be left behind or be troublesome to remove if not needed. It was decided to start from a blank project to avoid these problems. However, some of the example projects were created and investigated to help get a feel for Android studio.

While looking at the bottom navigation bar example, some key features were found that would be needed for the application. How to run a project on the emulation was one of the skills learn from investigating template projects. As well as learning how to debug and step line by line through an application. learning this was key to the development because being able to run and debug the program anytime would make sure that the application was as bug free as possible at all times.

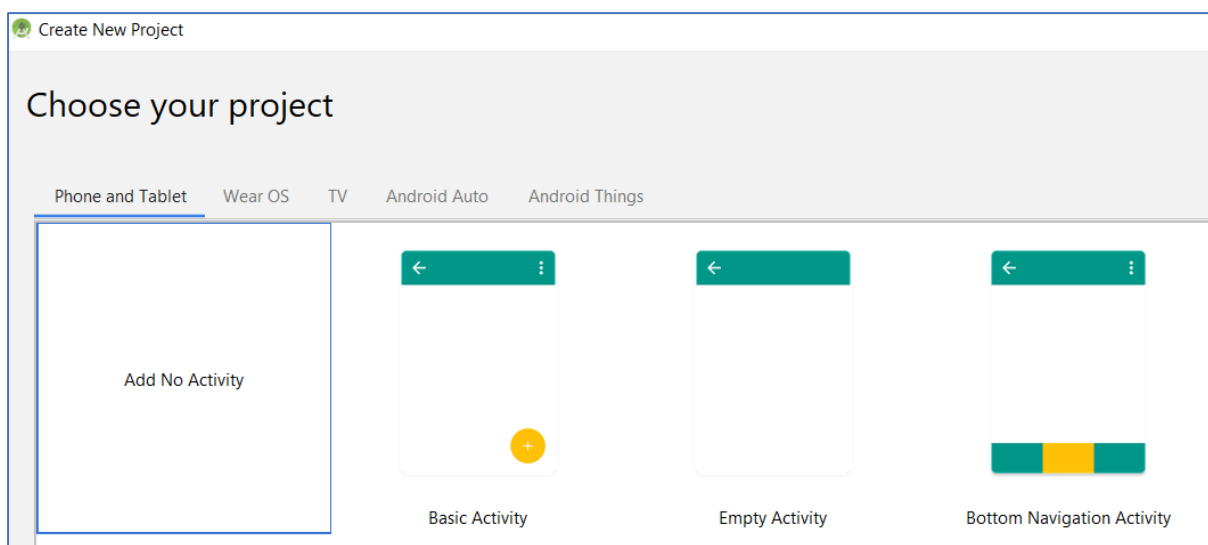


Figure 10 - Project examples page

Along with learning how to run the application, all the originally planned Java class files and layout files were created. This included all the main pages such as log in screens, task lists and account pages to name a few. While still looking at the template projects, the ability to link Java and Layout classes was learnt. This would give all the layout pages a java class to run the code needed for that page.

```
@Override
public void onCreate(@Nullable Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.login_layout);
}
```

Figure 11 - Code example, linking to login layout

Super and onCreate are Android functions used to call the code within the respective class. Without calling super.onCreate(savedInstanceState) the code within the class would not be called properly in the hierarchy of the application. After calling onCreate, a layout can be set using the setContentView then inputting the appropriate layout for the code being run.

5.3 Layouts

Soon after looking into layout files (XML),

When learning about layout (XML) files I quickly realised that this is where I need to make sure the designs accommodate different device sizes. To able to do this I researched into all the tools that I had access to in the layout editor of Android Studio. The layout editor provides access to several layout choices. Each of these give the user different tools for designing the page.

Linear layout gives the user a layout that is organised into a single horizontal or vertical row with automatic scrolling once the number of attributes in a single page exceeds the size of the device screen. The limitation to this layout is the developer has no control over the placement of attributes on the page, the attributes will always be in a single row with the only customisation being how close they are to each other.

Constraint layouts was the best option for designing the application. Constraint layouts give the user the ability to lock attributes to a user decided distance from another. This feature was used to be able to make sure none of my attributes fall outside of the user's screen size. The decision was made to use the constraint layout throughout all the layout designs. This would ensure that all the pages of the application would scale automatically to fit most screen sizes.

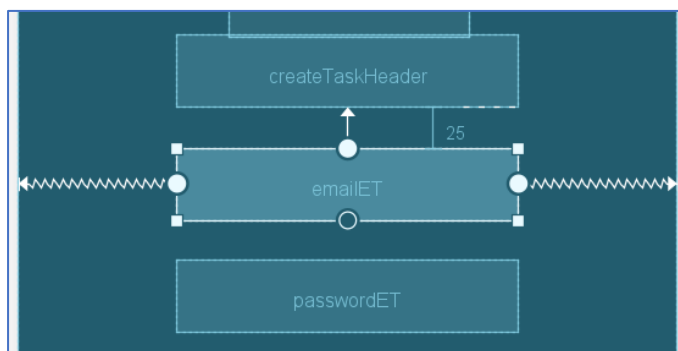


Figure 12 - Example image of an attribute being constraint

- Centred and 25 density-independent pixels (dp) from previous attribute

5.4 Functionality

5.4.1 Basic Activity's

Within Android studio, an Activity is the name given to a page of an application. Activities can hold a single layout and need to be linked by interacting with the application. the most common way to swap activities is to include buttons with code to transition to the next. Most of the applications pages could be made with activities and a single layout.

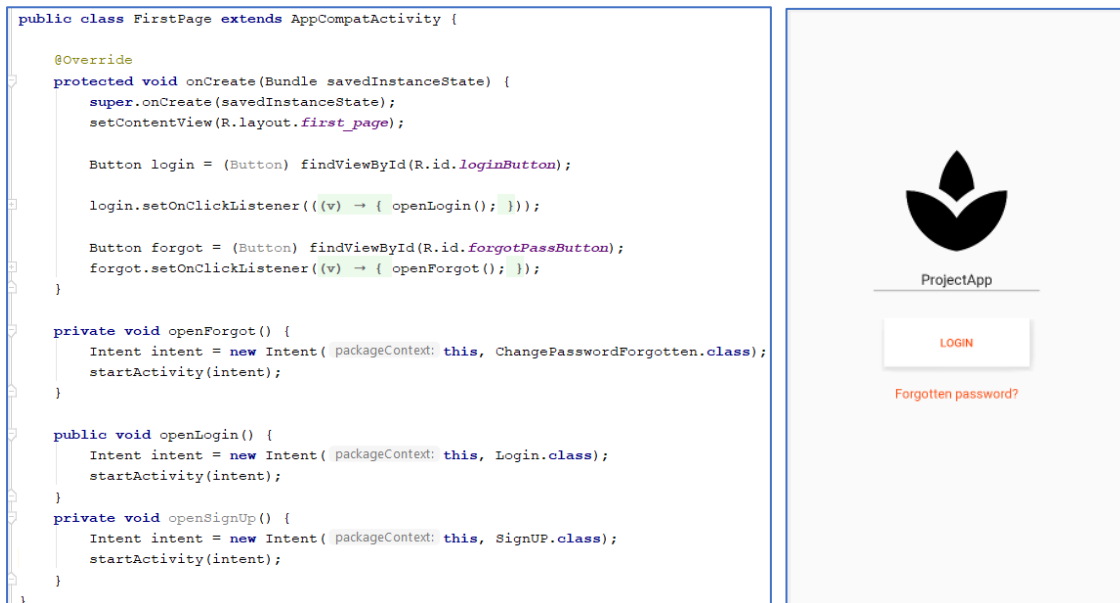


Figure 13 - Code and layout for the first page of the application

- The left screenshot shows the code needed for the first page to have button interactions
- The right screenshot shows the first page of the application

From the file structure laid out previously, all of the needed activities could be made in preparation for additional to be added later. At this point in development the application could move between activities with simple button commands. To create a button, the layout attribute needed to be linked to a declared variable within the class being used. Button “variable name” = `(Button)` declares the variables and is ready to use. The id of a layout attribute is then given to this declared variable and a listener can now be attached.

A listener is an Android function that constantly checks if the user has tapped a section of the layout that has the listener attached to it. The listener then runs a function to move to another page or any request needed for the given action.

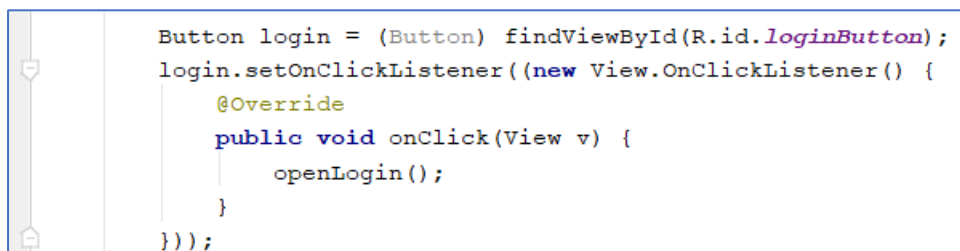


Figure 14 - Example of a listener function used throughout the project

To move to new activities, functions were created with the only intent was to move the application to the next page. By using `Intent`, the application could be given the class for what the button it was attached to was directing to. For example, the front pages log in button would receive the `openLogin()` function.

This approach was used throughout the application as it was a quick and simple method of making buttons within Android Studio.

```

public void openLogin() {
    Intent intent = new Intent( packageContext: this, Login.class);
    startActivity(intent);
}

```

Figure 15 - Example of a function used to move to a new activity

5.4.2 Bottom Navigation & Fragments

To be able to design a working navigation bar, the application would need to be able to use fragments. Fragments allow the application to display multiple different activities on one page. The use of fragments was beneficial to the project application as swapping between fragments is a faster process than swapping between multiple activities (Google, inc., 2020). By using fragments for the navigation bars meant that users can quickly swap between the important sections of the application.

In order to create a navigation bar, the application needed to use a manager class to swap between the fragments needed to run the main application pages. The manager class worked off waiting for a navigation icon to be pressed then directing the user to a new fragment for the appropriate selection.

The buttons on the navigation bar work the same way as describe previously to move between activities with only one difference, the listener is now handled by a Switch statement and is checking to see which button id was pressed. Once a button is pressed the Switch statement finds the same id and moves to the fragment associated with that id.

```

//creating a listener function to use for each navigation menu
private BottomNavigationView.OnNavigationItemSelectedListener navListener =
    new BottomNavigationView.OnNavigationItemSelectedListener() {
        @Override
        public boolean onNavigationItemSelected(@NonNull MenuItem navItem) { //menu item = the scene to swap to
            Fragment selectedFragment = null;

            switch (navItem.getItemId()) {
                case R.id.nav_account:
                    selectedFragment = new TeacherView_Account();
                    break;
                case R.id.nav_list:
                    selectedFragment = new TeacherView_TaskList();
                    break;
                case R.id.nav_add:
                    selectedFragment = new TeacherView_CreateTask();
                    break;
                case R.id.nav_adds:
                    selectedFragment = new TeacherView_CreateStudent();
                    break;
            }
            getSupportFragmentManager().beginTransaction().replace(R.id.nav_container,
                selectedFragment).commit();
            return true;
        }
    };

```

Figure 16 - Image of teacher's manager class code

- Section of code shows the switch statement to select the correct fragment depending on the icon pressed on the navigation bar.

Manager classes hold the layout for the navigation bars and all the icons needed to represent each main page of the application.



Figure 17 - Teachers navigation bar

- From left to right, Task list, Create Task, Create user, Account

Developing in fragments was difficult to say the least. Using fragments cause a lot of unexpected problems with the application that were eventually fixed. First was linking to the correct layout, additional steps were needed to able to link to the correct layout.

```
public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
    View v = inflater.inflate(R.layout.teacher_view_account_layout, container, attachToRoot: false);

    return v;
}
```

Figure 18 - Example of additional steps to give a layout

A View needed to be created and given the layout id, then at the end of the code written in this class, the View needed to be returned. This would give the current activity the correct fragment needed at that point in time.

Secondly, moving to new a new page (activity) the fragment needed to be given the current activities context. Without passing this data the application would crash immediately when clicking a button. The getActivity() function was needed in order to get the correct context.

```
Intent intent = new Intent(getActivity(), FirstPage.class);
```

Figure 19 - Fragment to new activity

Also, while working with fragments, using getActivity() was sometimes not enough to be able to get the correct activity context. Leading to additional crashes and needed to move up another level in the hierarchy of activities and fragments. To do this getApplicationContext() was needed. Finding this additional function was tricky and frustrating but once one problem was fixed the function could be tried and tested with other errors that popped up.

```
UserAccountManager.getInstance(getActivity().getApplicationContext()).logout();
```

Figure 20 - Additional fragment hierarchy example

5.4.3 Teacher and Student Views

Due to the differences already set out in the design plan, teacher and student activities will need to be created differently. The application was first developed around the teacher views as some of the pages present within this view will be shared with the student view. These pages included the task list and the account page.

A layout could be made for the teacher's task creation page from the drawn deign mock-ups. The task creation page was originally the only page that was exclusive to the teacher view this was because originally, the application gave all users the ability to create an account. This was handled by a page before login where the user would input their details and then select an option to be a teacher or student.

Figure 21 - Original first page & account creation page

By using the Radio buttons on the create account page, a fake login system was created to get the user to the appropriate teacher or student view. When radio buttons are grouped together, only one can be selected at a time.

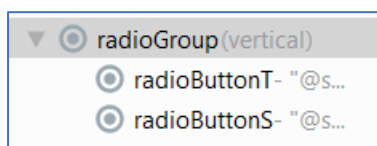


Figure 22 – Radio button grouping

Code was written to check to see if a button was checked. Then the corresponding teacher or student view was set as the Intent for the next activity inside of two if statements. This approach helped to navigate to a teacher or student view for early stages of testing. A similar method was used later in development when checking if the users account was a teacher or student from the information provided from the database.

```
private void findNextPage() {
    // this finds the next page. student or teacher depending on what radio button was selected.
    if (studentRadio.isChecked())
    {
        Intent intent = new Intent(getActivity(), StudentView_Manager.class);
        startActivity(intent);
    }
    if (teacherRadio.isChecked())
    {
        Intent intent = new Intent(getActivity(), TeacherView_Manager.class);
        startActivity(intent);
    }
}
```

Figure 23 - Radio button usage when creating an account

- A different Intents was given depending on the radio button selected

This page however was flawed by design. Giving the students the ability to create their accounts could lead to numerous troubles including getting the wrong teacher id or signing up as a teacher. With further research, the decision was made to move the account creation page to the teachers view. Giving them the ability to make new teacher and student accounts.

The differences between the two views was not much of an issue than first anticipated. The main differences cause by the separation of views were the navigation bar and how the data in the task list was accessed. Both problems were solved quite easily with minimal changes to the code and design. The student navigation bar constantly shows the name of the pages because there is already less information to show and to aid in the student learning the 2 buttons.



Figure 24- Comparison of Teacher and Student navigation bars

- Left image is the final Teacher navigation bar
- Right image is the final Student navigation bar

At this stage of development, the application consisted of a front page prompting to log In or create an account, a fake account creation page to transition the user to the teacher or student view depending on the selected option. Teacher and student views with different navigation bars. And finally, a task creation page and account pages.

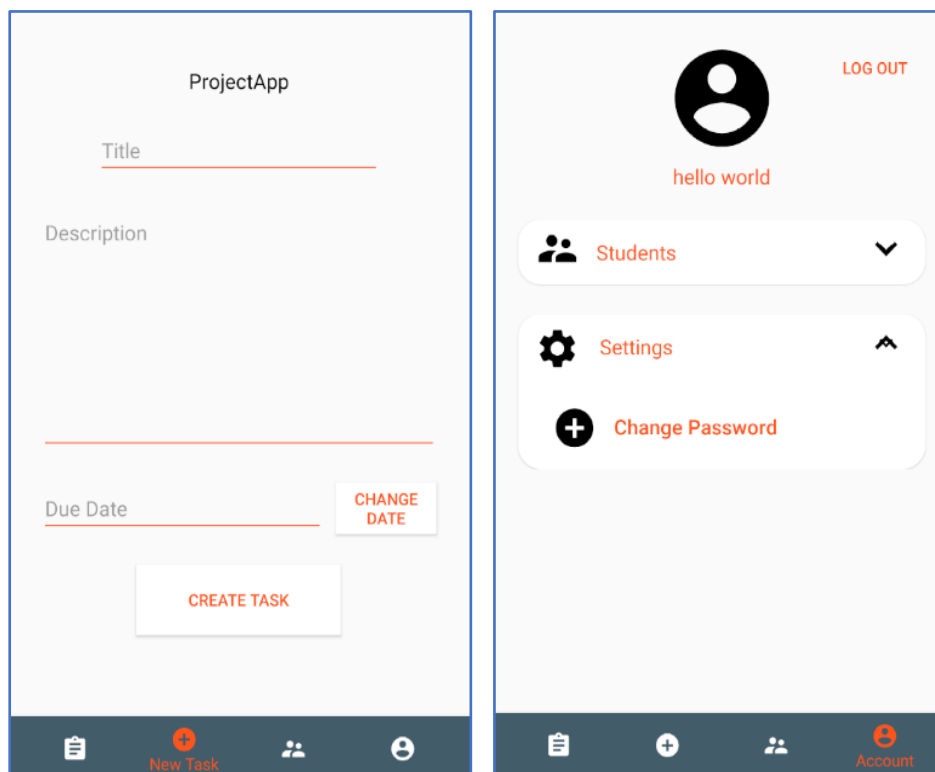


Figure 25 - Images of Create Task & Account pages

- Left image – the design of the task creation page, made using Editable text boxes with the same constraining layout style as described before.
- Right image – teachers Account page with an expandable card button the when clicked reviles a change password button.

5.4.5 Database Connectivity Using php

At this stage of development, the basics of an application were completed. The app could navigate correctly around all the pages designed without crashing or slowdown, even while running on a less reliable emulator.

Before starting development of the php code, Xampp needed to be set up and the PhpMyAdmin created. Simply opening Xampp and turning on Apache and MySQL services gives access to the local PhpMyAdmin database via a simple link inserted into any browser.

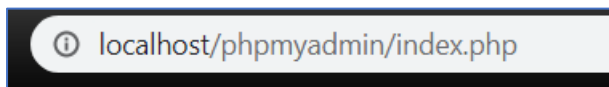


Figure 26 - Localhost link

The next stage of development was to be able to access the database via Php code. This would mean the user would be able to create accounts, tasks and view all the content needed. Starting work on the Php was difficult at first. Constant variables were declared for the connection data of the database.

```
Includes > Constraints.php > ...
1  <?php
2      define('DB_NAME', 'fyp');
3      define('DB_USER', 'root');
4      define('DB_PASSWORD', '');
5      define('DB_HOST', 'localhost');
```

Figure 27 - Php constants

Using the defined constants, a new class was made to allow the database to be logged in to. Creating this file meant that this could be used within all other operations to avoid writing the same code in each file to initially connect to the database. This file reads in the constants and inserts them into a mysqli __construct() query. This query uses that data to login to the requested database.

```
Includes > DbConnect.php > DbConnect
1  <?php
2  class DbConnect{
3      private $con;
4      function __construct(){
5      }
6      function connect(){
7          include_once dirname(__FILE__).'/Constraints.php';
8          $this->con = new mysqli(DB_HOST, DB_USER,
9                                DB_PASSWORD, DB_NAME);
10
11          if(mysqli_connect_errno()){
12              echo "Failed to connect";
13          }
14          return $this->con;
15      }
16  }
```

Figure 28 - Php connect

The php code was being written within Visual Studio Code, a lightweight version of Visual Studio. Php files cannot be run normally by computers and a third-party application needed to be used to help run the files for testing. Postman is a HTTP client used for testing web services. This software was used as it is one of the more popular and easy to use software for the testing that needed to be done (Postman, inc., 2020). Postman was especially helpful later in development when needing to POST to the database. The software allows the user to insert data as if it was being received from another source. Also, by using this software testing of the php code could be done without the need to incorporate it into the Android application.

A similar approach of using constant values was used in the application. Assigning links for each database query to variables made calling the correct query easier when developing the Android application code. A very simple solution was used when connecting the application to the internet, that being using the ip address of the computer being worked on. However, such a simple solution came with a drawback of needing to be change when moving computes or Wi-Fi. For further development of the application this would need to be changed to dynamically get the uses ip address so the connection to the internet can happen.

```
public class pHp_Constants {
    private static final String ROOT_URL = "http://192.168.0.16/Android/WebServices/";

    static final String URL_REGISTER = ROOT_URL + "RegisterUser.php";
    static final String URL_LOGIN = ROOT_URL + "UserLogin.php";
    static final String URL_CREATETASK = ROOT_URL + "CreateTask.php";
    static final String URL_GETTASKS = ROOT_URL + "GetTasks.php"; //old attempt
    static final String URL_GETUSERTASKS = ROOT_URL + "GetUserTasks.php";
    static final String URL_UPDATEPASS = ROOT_URL + "UpdatePass.php";
    static final String URL_DELETEUSER = ROOT_URL + "DeleteUser.php"; //old attempt
    static final String URL_DELETETASK = ROOT_URL + "DeleteTask.php";
    static final String URL_UPDATETASK = ROOT_URL + "UpdateTask.php";
    static final String URL_ISCOMPLETED = ROOT_URL + "UpdateTaskCompleted.php";
}
```

Figure 29 - Android Php constants

5.4.6 Account Creation and Login

To send data to the database, information needed to be sent using a POST request. Using POST would read in the data that is being given (the users account details in this case) and insert them into a query. Creating a file called RegisterUser.php, testing the insertion of data could now happen via Postman. This file runs two if statements before accepting any data. First if the method of sending that data is a POST and if all the data that I want to collect has values. By doing this it means that users cannot accidentally press create account early and send unfinished data to the database.

```

WebServices > RegisterUser.php > ...
1  <?php
2  require_once '../includes/DbOperations.php';
3
4  $response = array();
5  if($_SERVER['REQUEST_METHOD']=='POST'){
6      if(
7          isset($_POST['firstname']) and
8          isset($_POST['lastname']) and
9          isset($_POST['email']) and
10         isset($_POST['password']) and
11         isset($_POST['isStudent']) and
12         isset($_POST['accOwner']))
13     {

```

Figure 30 - RegisterUser.php - data checking

The second half of the file pushed the data to the query ready to be prepared to be moved to the database. Here is where Postman was especially helpful. The program allowed data to be inputted as if it was being sent from the application. This always allowed testing to be done for all of the requests needed to be made.

```

14 //operate data futher
15 $db = new DbOperations();
16 $results = $db->createUser($_POST['firstname'],
17                             $_POST['lastname'],
18                             $_POST['email'],
19                             $_POST['password'],
20                             $_POST['isStudent'],
21                             $_POST['accOwner'],
22                             );

```

| | KEY | VALUE |
|-------------------------------------|-----------|-------|
| <input checked="" type="checkbox"/> | firstname | Hello |
| <input checked="" type="checkbox"/> | lastname | World |

Figure 31 - RegisterUser.php - pushing data to query

- Bottom Image shows an extract of inserting data via postman.

Before inserting the data to the database, the password the users have provides is encrypted using MD5 hashing. MD5 hashing creates an encoded string of data from the given data. MD5 is “almost impossible” to reverse, making it a quick and easy way to include password encryption in to the prototype of the application.

Depending on the query depended on what that data was used for. In this example of creating a user, the query needing to be used was INSERT. The Prepare function is where the selected query is chosen. INSERT will take the VALUES from the RegisterUser.php and matches them with corresponding table id. FirstName in the INSERT query is the name within the database table, these

must match, or the query will fail. With these matching and the query pointing to the 'users' table, the inserted data from Postman can now be applied to the database.

```

11  /* create a record */
12  public function createUser($firstname,$lastname,$email,$pass,$isStudent,$accOwner){
13      if($this->checkDuplicate($email)){
14          return 0;
15      }else{
16          $passwordENC = md5($pass); /*encrypt password*/
17          $stmt = $this->con->prepare("INSERT INTO `users` (`id`, `FirstName`, `LastName`, `Email`, `Password`, `isStudent`, `AccOwner`)
18                                  VALUES (NULL, '$firstname', '$lastname', '$email', '$passwordENC', '$isStudent', '$accOwner')");
19
20          $stmt->bind_param("sssss",$firstname,$lastname,$email,$passwordENC,$isStudent,$accOwner);

```

Figure 32 - Creating a user INSERT query

Next was creating a php function to log the users in to the database. This needed to be broken down in to two parts, providing email and password of an account then acquiring all the data associated with a user's email and storing. Getting the users email and password was simple. The method used the same POST request as with the account creation but with only their email and password.

```

6      if($_SERVER['REQUEST_METHOD']=='POST'){
7          if(isset($_POST['email']) and isset($_POST['password'])){}

```

Figure 33 - Login POST

A SELECT query was then used to get the id number of the user with match credentials.

Bind_param() is a required function when creating a mysqli query, this function prepares the given variables to the statement, allowing them to then be used in the execute() function. If the inputted data matched what was in the database, the function would return num_rows > 0 and log the user in.

```

44  public function userLogin($email,$pass){
45      $passwordENC = md5($pass);
46      $stmt = $this->con->prepare("SELECT id FROM users WHERE Email = ?
47                                  AND Password = ? ");
48      $stmt->bind_param("ss",$email,$passwordENC);
49      $stmt->execute();
50      $stmt->store_result();
51      return $stmt->num_rows > 0;
52  }

```

Figure 34 - SELECT query checking user login data

Now that the user is successfully logged in, the additional data connected to the account needed to be acquired. This was done by using a SELECT query again and using * as a shortcut to collect from all the columns in the database. The results from the database table are now fetched and applied to an array called \$user using get_results()->fetch_assoc();

```

54  public function getUserByEmail($email){
55      $stmt = $this->con->prepare("SELECT* FROM users WHERE Email = ?");
56      $stmt->bind_param("s",$email);
57      $stmt->execute();
58      return $stmt->get_result()->fetch_assoc();
59  }

```

Figure 35- Example of SELECT* and fetch_assoc() in user login

The final step was to convert the array of data into a usable format. The format chosen to use was a JSON object. JSON was chosen because it is a very easy to use format web trying to communicate with the web. JSON objects are broken down into keys and data, these were easy to read inside of Android Studio, making the process of getting the php functions much easier.

```

12      $user = $db->getUserByEmail($_POST['email']);
13      $response['error'] = false;
14      $response['id'] = $user['id'];
15      $response['firstname'] = $user['FirstName'];
16      $response['lastname'] = $user['LastName'];

```

Figure 36 - Extract of applying fetched data to a JSON object

- The variable \$response is the JSON object receiving the data

Before moving on to creating additional database functions, including the working account creation and login functions into the Android application was the next task. To be able to achieve a fast working system, the Volley library was needed. Volley is a HTTP library inside of Android Studio designed to connect and bridge a gap between application and internet (Google, inc., 2019). Volley is now an older HTTP library with an alternative boasting to be faster called Retrofit (Square, Inc., 2020). However, Retrofit is not included with Android Studio and therefor, the guides and documentation are not tailored to Android Studio unlike Volley.

Implementing the RegisterUser query was the first on the to do list. The query was tested extensively before being implemented. Knowing that there were zero bugs with code was a time saver when figuring out how it all worked inside of Android Studio.

First, the user inputted data on the create account page needed to be read. To do this the edit text boxes from the layout file were allocated variables in the class file, then any text in the box when the create account button was taken and converted into a String. This was done for all appropriate database columns, except the account owner. This was to be added once the login system was implemented and the user account manager was working.

```

        firstName = (EditText) v.findViewById(R.id.firstNameET);
        lastName = (EditText) v.findViewById(R.id.lastNameET);

        private void registerUser(){
            //reading the data from the text boxes and putting them in to strings so they can be sent the database
            final String firstNameSr = firstName.getText().toString().trim();
            final String lastNameSr = lastName.getText().toString().trim();
            final String emailSr = email.getText().toString().trim();
            final String passwordSr = password.getText().toString().trim();
            final String isStudentSr = isStudent.getText().toString().trim();
            final String accOwnerSr = UserManager.getInstance(getActivity().getApplicationContext()).getUserEmail();

```

Figure 37 - Reading text inputted by the user

Next, the newly created Strings made from the users inputted data is used inside of a Volley String Request. This string request takes a HashMap as a parameter and this is where the data is sent to the database. The HashMap is allocating the newly made strings to the appropriate database entry made in the php script. In the php scripts, the users first name was POSTed to a variable called "firstname". This variable is being filled with the corresponding user inputted data just as before with the example using Postman.

```

@Override
protected Map<String, String> getParams() throws AuthFailureError {
    Map<String, String> params = new HashMap<>();
    params.put("firstname", firstNameSr);
    params.put("lastname", lastNameSr);
    params.put("email", emailSr);
    params.put("password", passwordSr);
    params.put("isStudent", isStudentSr);
    params.put("accOwner", accOwnerSr);
    return params;
}

```

Figure 38 - HashMap for Registering a user

- "firstname" = php script variable
- firstNameSr = user data as a string

Once this data has been allocated, the php script is ran on the webservices of Xampp and passed to the database. If the query was successful, then a JSON object will be returned. This JSON object includes a message to the user to inform them if the request was successful or not.

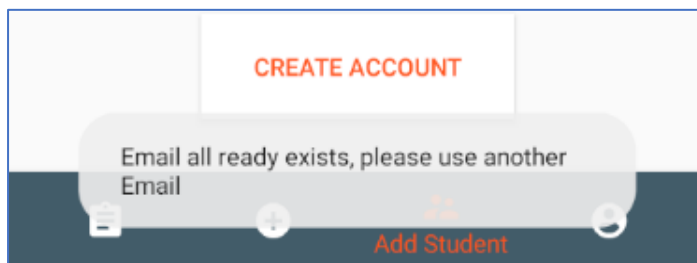


Figure 39 - Example of JSON object message

A full example of the registering a user String request can be found in Appendix E.

Logging the user into the application was the next step. The php script had already been made and tested, but now the data received from that script needed to be stored inside of the application. This is needed so some of the data can be used, such as displaying the user's name on the account page or correctly associating the owner of a task.

The log in process started the same as the registering users. Creating strings to read in the user's email and password and then applying them to a HashMap so they can be POSTed via the php script.

```

@Override
protected Map<String, String> getParams() throws AuthFailureError {
    Map<String, String> params = new HashMap<>();
    params.put("email", emailSr);
    params.put("password", passSr);
    return params;
}

```

Figure 40 - Login HashMap

The string request then proceeded as normal, upon reaching the JSON object response. This is where the differences started. The JSON response for logging in included all the data about that user. This JSON was then separated into the appropriate "keys" and passed to a class storing the user's data.

A JSON key is one part of a pair. JSON keys must be strings and be paired with values. These values are associated with the key and can be read by other programs just by looking at the JSON key.

```
1 [{"error":false,"id":18,"firstname":"hello","lastname":"world","email":"@email",
  "password":"202cb962ac59075b964b07152d234b70","isStudent":"false","accOwner":""}]

JSONObject obj = new JSONObject(response);
if (!obj.getBoolean( name: "error")){
    UserManager.getInstance(getApplicationContext())
        .userLogin(obj.getInt( name: "id"),
                    obj.getString( name: "firstname"),
                    obj.getString( name: "lastname"),
                    obj.getString( name: "email"),
                    obj.getString( name: "password"),
                    obj.getString( name: "isStudent"),
                    obj.getString( name: "accOwner")
                );
    findAccountType(obj);
}
```

Figure 42- Logging in JSON response

- keys are displayed with quotation marks inside of a JSON object.

The JSON values are assigned to a string in Android studios shared preferences and stored using keys. These keys would hold all data for on attribute from the database. Example the users first name would be stores in KEY_USER_FIRST. these new android keys could be used to request the string when ever needed in a different part of the application.

```
//additional functions to login and store the users data
public boolean userLogin(int id, String firstName, String lastName, String email,String password, String isStudent, String accOwner){
    SharedPreferences sharedPreferences = ctx.getSharedPreferences(SHARED_PREF_NAME, Context.MODE_PRIVATE);
    SharedPreferences.Editor editor = sharedPreferences.edit();

    editor.putInt(KEY_USER_ID, id);
    editor.putString(KEY_USER_FIRST, firstName);
    editor.putString(KEY_USER_LAST, lastName);
    editor.putString(KEY_USER_EMAIL, email);
    editor.putString(KEY_USER_PASSWORD, password);
    editor.putString(KEY_USER_ISSTUDENT, isStudent);
    editor.putString(KEY_USER_ACCOWNER, accOwner);
    editor.apply();
    return true;
}

String getUserName(){
    SharedPreferences sharedPreferences = ctx.getSharedPreferences(SHARED_PREF_NAME, Context.MODE_PRIVATE);
    return sharedPreferences.getString(KEY_USER_FIRST, defValue: null);
}
```

Figure 41- User account manager class

obj.getString("isStudnt") is then check if the value if true or false. If the value received from the JSON object was false, the user would be directed to the teacher view of the application. with True

sending them to the student view.

```
private void findAccountType(JSONObject obj) throws JSONException {  
  
    if (obj.getString( name: "isStudent").equals("false")){  
        Intent intent = new Intent( packageContext: this, TeacherView_Manager.class);  
        startActivity(intent);  
    }  
    else if(obj.getString( name: "isStudent").equals("true")){  
        Intent intent = new Intent( packageContext: this, StudentView_Manager.class);  
        startActivity(intent);  
    }  
}  
}
```

Figure 43 - Application checking if the users is a teacher

The shared preference manager could be used again to log the user. A php script was not needed to do this. The application would clear all the data stored in the shared preference manager, leaving it empty so another user could log in.

```
public boolean logOut(){  
    SharedPreferences sharedPreferences = ctx.getSharedPreferences(SHARED_PREF_NAME, Context.MODE_PRIVATE);  
    SharedPreferences.Editor editor = sharedPreferences.edit();  
    editor.clear();  
    editor.apply();  
    return true;  
}
```

Figure 44 - Log out function

5.4.7 Task Creation & Task Viewing

Creating a task used an identical method to create a user, with only slight differences. The php script handling the POST to the database linked to the tasks table instead of the users table.

```
29 public function createTask($tasktitle,$taskdesc,$taskowner,$taskdate,$taskcompleted){  
30  
31     $stmt = $this->con->prepare("INSERT INTO `tasks` (`id`, `TaskTitle`, `TaskDescription`,  
32         `TaskOwner`, `TaskDate`, `TaskCompleted`)  
33         VALUES (NULL, '$tasktitle', '$taskdesc',  
34             '$taskowner', '$taskdate', '$taskcompleted')");  
35  
36     $stmt->bind_param("sssss",$tasktitle,$taskdesc,$taskowner,$taskdate,$taskcompleted);
```

Figure 45 - INSERT query for task creation

The user would only input 3 section of a task, title, description and due date. The task owner was handled by the user account manager by getting the users email address and filling in that string. The check of the task being completed was always set to false and not completed by default.

```
String getUserEmail(){ // main use in TEACHER task list  
    SharedPreferences sharedPreferences = ctx.getSharedPreferences(SHARED_PREF_NAME, Context.MODE_PRIVATE);  
    return sharedPreferences.getString(KEY_USER_EMAIL, defValue: null);  
}
```

Figure 46 - Inputting the task owner

Before pulling data from the database, some layouts needed to be made so the user can see the tasks. Each task pulled from the database would be assigned to a task card and added to the RecyclerView.

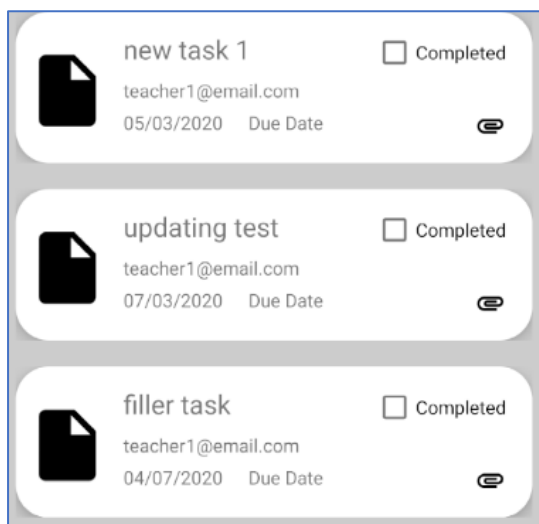


Figure 47 - RecyclerView of Task cards

Displaying the tasks information was much more complicated and cause lots of issues with development. trying to get this feature to work cause weeks of delays then eventually came down to small spelling errors in the php script and attempting to implement it inside of a fragment. To get this to work, the php script needed to be removed from all over functions and written in a single file. This broke the object orientated programming approach that was being used for all other php scripts. The need to make the file sperate doesn't slowdown the connection to the database but is annoying that this one function is separated from the rest.

In appendix E 2 , the commented-out code (green text) shows that there was a check to see who that task owner was while inside of the php query. This old approach would work faster than the one that got implemented but due to problems in Android studio, a second solution was used. The new approach did not need a POST request for the php query. The query used SELECT* to pull all the information from all rows within the task table. The results were then pushed into an array and echoed as a JSON response.

```

17     $stmt = $con->prepare("SELECT* FROM tasks");// WHERE TaskOwner = ?");
18     //$stmt->bind_param("s",$taskowner);
19     $stmt->execute();
20     $stmt->bind_result($id, $TaskTitle,$TaskDescription,
21                     $TaskOwner,$TaskDate,$TaskCompleted);
22     $tasks = array();
23
24     while($stmt->fetch()){
25         $temp = array();
26         $temp['id'] = $id;
27         $temp['TaskTitle'] = $TaskTitle;
28         $temp['TaskDescription'] = $TaskDescription;
29         $temp['TaskOwner'] = $TaskOwner;
30         $temp['TaskDate'] = $TaskDate;
31         $temp['TaskCompleted'] = $TaskCompleted;
32         array_push($tasks, $temp);
33     //}
34 }

```

Figure 48 - Task collecting array

Now that the tasks had been read from the database, the information now needed to be displayed to the user. A task manager class was made and worked in the same ways as the user account manager. Taking the data from the JSON response and storing it to be used later. Since the JSON response was an array of data, a JSONArray was needed to be able to run through all the data correctly and pull out each task from the collection of tasks.

```
JSONArray tasksJSON = new JSONArray(response);
for(int i =0; i<tasksJSON.length(); i++){
    JSONObject tasksObject = tasksJSON.getJSONObject(i);
    int id = tasksObject.getInt( name: "id");
    String title = tasksObject.getString( name: "TaskTitle");
    String description = tasksObject.getString( name: "TaskDescription");
```

Figure 49 - JSONArray for collecting all tasks

Because the php query did not already filter out the tasks by the correct owner, the application needed to do so. This was an easy process and involved comparing to strings, if they matched then add the task to the list about to be displayed. Once a task was added to the task list, the data pulled from the JSON object was passed to an Adapter.

```
Tasks task = new Tasks(id,title,description,owner,date,completed);
String temp = task.getTaskOwner();
if(temp.equals(TaskOwner)){
    tasksList.add(task);
}
}
adapter = new TaskAdapter(getActivity(), tasksList);
recyclerView.setAdapter(adapter);
adapter.setOnItemClickListener(TeacherView_TaskList.this);
```

Figure 50 - Checking for the owner of a task

- adapter = new TaskAdapter Is passing the data to the adapter ready to be used in a ui element

Android adapters work by connecting data and ui elements together. By assigning where the data needs to be, the adapter puts the correct data in to the correct ui element (Google, inc., 2019). This was especially useful as this can be done often without needing to store the data, so populating the recycler view of tasks was handled quickly and easily. Appendix E 3 shows a full view of the adapter associating the data to the tasks card elements.

Finally, for displaying the tasks was to have them be clickable to expand and show the description. Since the description could not be seen in the card view of the task, a way to expand the card was needed. This feature could be used for more than just the description but due to time constraints, only the description was added. Expanding a task card to a full page required changing fragments. To do this the data that was needed for that page needed to be passed via an Intent, the same way a button works to move to a new page.

```

@Override
public void OnItemClick(int pos) {
    Intent intent = new Intent(getActivity(), Task_StudentView.class);
    TasksManager clickedItem = tasksManagerList.get(pos);

    intent.putExtra(ID, clickedItem.getIdString());
    intent.putExtra(TITLE, clickedItem.getTaskTitle());
    intent.putExtra(DESCRIPTION, clickedItem.getTaskDesc());
    intent.putExtra(DUEDATE, clickedItem.getTaskDate());
    intent.putExtra(ISCOMP, clickedItem.getTaskComp());

    startActivity(intent); //sending the caught data to the task activity
}
}

```

Figure 51 - Passing data via Intent to expanded task page

Teachers and students received different expanded task pages. Teachers had a page that allowed them to update the task by simply changing the text in the edit text boxes, the same way they did when creating the task. Along with the ability to delete the task. Students can check off if the task was completed or not. Looking back, teachers should have had the ability to check off tasks as well. This would leave the teacher change the task completion if they felt the student had not finished correctly.

| | |
|---|--|
| <p style="text-align: center;">new task 1</p> <hr/> <p>this is the first task</p> <hr/> <p>05/03/2020</p> <div style="text-align: center;"> <p>UPDATE TASK</p> <p>DELETE TASK</p> </div> | <p style="text-align: center;">new task 1</p> <p>this is the first task</p> <p>05/03/2020</p> <p>Completed? <input type="checkbox"/> false</p> |
|---|--|

Figure 52 - Teacher and student views of a full task

- Left image is of the teacher view of an expanded task
- Right image is of the student view of an expanded task

5.4.8 - Update and Delete

When learning to update and delete entries from a database, a good understanding of php and MySQL queries was established which resulted in a shorter period to include these additional features. When making the expanded task view, the id of the task needed to be collected to help with the selecting process when the user clicked. This id was then used to help update the task.

Updating a task POSTed data like other function before but used an UPDATE query to change the data that the user requested to change. The difficulty of updating a task came from needed to send the correct id number of the database entry.

```
8
9     $result = $db->updateTask($_POST['tasktitle'],
10                               $_POST['taskdesc'],
11                               $_POST['taskdate'],
12                               $_POST['id']);
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88 public function updateTask($tasktitle,$taskdesc,$taskdate,$id){
89     $stmt = $this->con->prepare("UPDATE tasks SET TaskTitle = ?, TaskDescription = ?,
90                                TaskDate = ? WHERE id = ?");
91     $stmt->bind_param("sssi", $tasktitle,$taskdesc,$taskdate,$id);
92     if( $stmt->execute()){
93         return true;
94     }else{
95         return false;
96     }
97 }
```

Figure 53 - Updating a task php

Again, a similar approach was used to create an updating password system. The user would insert the new password they wanted and that will be POSTed via the UPDATE query. the updatingPass function SETs the password in WHERE the email is the user's email, pass from the user account manager.

```
77 public function updatePass($pass, $email){
78     $passwordENC = md5($pass);
79     $stmt = $this->con->prepare("UPDATE users SET Password = ? WHERE Email = ?");
80     $stmt->bind_param("ss", $passwordENC, $email);
81     if( $stmt->execute()){
```

Figure 54 - UpdatePass php function

This update function was used for a change password page and a forgotten password page. The differences between these pages were the change password page would check the user inputted their origin password first before updating the database. However, this proved difficult due to MD5 encryption. Because the application received the encrypted password, the user would have to insert the encrypted password for the check to pass. An attempt to use MD5 encryption inside of Android studio to try and get this working, but development had to move on due to timing. The attempt at MD5 encryption in android can be seen in Appendix E 4.

Change Password

Old Password

New Password

Retype Password

CHANGE PASSWORD

Change Password

Email

New Password

Retype Password

CHANGE PASSWORD

Figure 55 - Updating password pages

Deleting a task was the easiest query to make. Again, the same procedure was used with all other functions except the DELETE query was used instead. By attaching the delete task button to the expanded task view, the application already had access to the id required for the php query.

```

119 public function deleteTask($id){
120     $stmt = $this->con->prepare("DELETE FROM tasks WHERE id = ?");
121     $stmt->bind_param("i", $id);
122     if( $stmt->execute()){
123         return true;
124     }else{
125         return false;
126     }
127 }

```

Figure 56 - DeleteTask function in php

However, there was a situation where the entire task database was deleted by accident because the Query was first designed without the WHERE id = ?. This meant that when any of the delete task buttons was pressed, it deleted all the tasks.

5.5 Database

5.5.1 Local and University

When setting up the application to run off a database, the original plan was to have it running from the university hosted phpMyAdmin database. By connecting to the university provided database, I would be showing an increased skill of reaching out and connecting to third party providers.

However, I did not manage to connect to the university database, while inside and outside of the university's Wi-Fi. After contacting the university IT support team, I was prompted to follow some

potential solutions. These included changing the port the server was running on to and resetting my Hompages account setting. Neither solution worked and my application still presented me with a Java security error, as shown below.

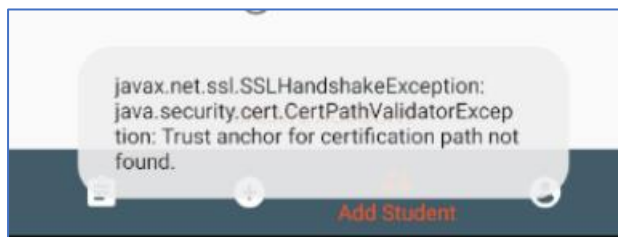


Figure 57 - Image of java security error on app

To work around not being able to connect to the university server, I created my own locally hosted database. This meant that I could easily connect to the database still via the internet, showing the skills of passing the data via the web.

5.5.2 Xampp set up

Xampp is a non-profit Apache web server installer. The program is designed to make the process of installing Apache, the world's most widely used web server service, easier and more manageable (The Apache Software Foundation, inc., 2020). I came across Xampp when researching in to how to set up my own locally hosted phpMyAdmin database. I knew I wanted to use phpMyAdmin so finding a software that works well with it was key. Xampp is the most popular php development environment and upon installation gives the user an option to create a phpMyAdmin database (Apache Friends, inc. , 2020).

After installation, Xampp created local folders within my computers C drive. To be able to use my php files for connecting to the database from the Android app, I needed to create a folder inside of Xampp's htdocs folder. Without locating the files here, they would not be able to interact with the Apache web server, rendering them useless. Because the php files needed to be in a specific folder inside of Xampp, this caused problems with the projects repository. I needed to manually make a copy of my php files and input them to a separate folder inside of the repository. This was only a minor annoyance but did cause me to forget to move the new php files to the repository sometimes when moving computer.

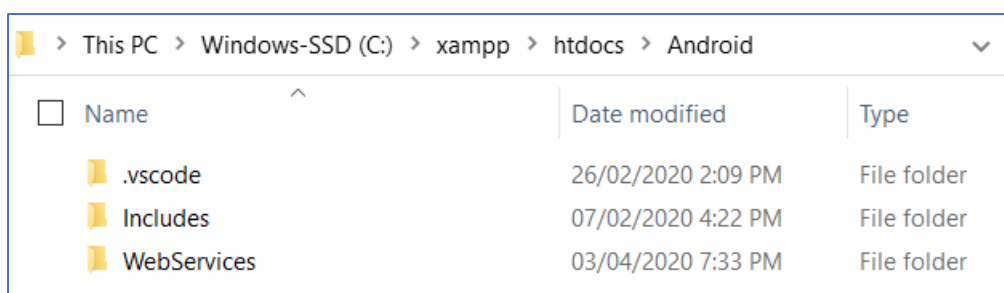


Figure 58 - Image of file structure for Xampp

5.5.3 Users

One the setup of my locally hosted database was completed; I could work on creating the tables that I would need order to store the user's information. The users table was created to store the user's id number, name, email, password, if they were a student and who owned the account.

- The users name would be displayed on their account page, showing to the user that it was there account that they had logged into.
- The email address and password were used as a log in system. The user would have to use their assigned email address and password to log in to the app.
- IsStudent was my method of checking if a user was a teacher or a student. Upon logging in, the application would check if the user was a student and move them the appropriate page.
- AccOwner was used when displaying the users task list. Upon loading the task list page, the application will check to see who owned the account and display all the tasks associated with that owner.

I chose to design the user table this way as it would give me all the information needed to make a user account for work tracking. The original design of the user table didn't include the AccOwner as I designed the log in system before tackling the database driven task list. Including this additional row to the table was easy with only minor changes to the old testing data.






















| + Options | | | | | | | | | | |
|--|---|---|---|----|-----------|----------|--------------------|----------------------------------|-----------|--------------------|
| <div><div></div><div></div><div></div></div> | | | | id | FirstName | LastName | Email | Password | isStudent | AccOwner |
| <input type="checkbox"/> |  |  |  | 18 | hello | world | @email | 202cb962ac59075b964b07152d234b70 | false | |
| <input type="checkbox"/> |  |  |  | 29 | teacher | 1 | teacher1@email.com | 202cb962ac59075b964b07152d234b70 | false | @email |
| <input type="checkbox"/> |  |  |  | 30 | teacher | 2 | teacher2@email.com | 202cb962ac59075b964b07152d234b70 | false | @email |
| <input type="checkbox"/> |  |  |  | 31 | student | 1 | student1@email.com | 202cb962ac59075b964b07152d234b70 | true | teacher1@email.com |
| <input type="checkbox"/> |  |  |  | 32 | student | 2 | student2@email.com | 202cb962ac59075b964b07152d234b70 | true | teacher1@email.com |
| <input type="checkbox"/> |  |  |  | 33 | student | 3 | student3@email.com | 202cb962ac59075b964b07152d234b70 | true | teacher1@email.com |
| <input type="checkbox"/> |  |  |  | 34 | student | 4 | student4@email.com | 202cb962ac59075b964b07152d234b70 | true | teacher2@email.com |

Figure 59 - Image of user's table from the database

5.5.4 Tasks

Creating the tasks table was more complicated than the users table. This was because I needed to find a way to give students their own individual tasks based on who their teacher was. The table consisted of 6 rows: id number, the tasks title, tasks description, the owner, due date and if the task was completed.

- The id number of the task is automatically generated when a new entry is added to the table. At the time of creation, this was not needed but I ended up using the id later in development to show the full task when clicked.
- The task tile would be displayed on the list of tasks to quickly see the task the user was looking for.
- The description of the title would be hidden on the task list be fully visible once the task in the list had been clicked.
- TaskOwner was created to show what teacher owned the task in the case that I could make multiple teachers linked to the same child account. This would also be used to compare with the student accounts own to fetch the correct tasks for each student
- The task due date is displayed to let the students know when the task is due
- TaskCompleted was implemented for students to be able to check off the task when finished.

The task table was harder to design than the user table. Coming up with a solution to be able to assign task to some students without having to make a new database table per student was a challenge that I had to overcome.

| | | | | | id | TaskTitle | TaskDescription | TaskOwner | TaskDate | TaskCompleted |
|--------------------------|------|------|--------|--|----|---------------------|---|--------------------|------------|---------------|
| <input type="checkbox"/> | Edit | Copy | Delete | | 21 | new task 1 | this is the first task | teacher1@email.com | 05/03/2020 | false |
| <input type="checkbox"/> | Edit | Copy | Delete | | 22 | updating test | task used for the updating test | teacher1@email.com | 07/03/2020 | false |
| <input type="checkbox"/> | Edit | Copy | Delete | | 23 | filler task | a task created to fill the recycler view to test t... | teacher1@email.com | 04/07/2020 | false |
| <input type="checkbox"/> | Edit | Copy | Delete | | 24 | filler task | a task created to fill the recycler view to test t... | teacher1@email.com | 04/07/2020 | false |
| <input type="checkbox"/> | Edit | Copy | Delete | | 25 | filler task | a task created to fill the recycler view to test t... | teacher1@email.com | 04/07/2020 | false |
| <input type="checkbox"/> | Edit | Copy | Delete | | 29 | second teacher task | a task testing the second teacher account displayi... | teacher2@email.com | 27/09/2020 | false |

Figure 60 - Image of tasks table from the database

5.6 Debugging

5.6.1 Application Problems

The debugging process with Android studio took a long time. Debugging slow and finicky and even worse while using an emulator at the same time. When the application would crash, most of the time no error message was displayed. Making the process very frustrating. A lot of the application was built inside fragment, and this is where a lot of the problems with the application came from. As briefly stated before, when inside of a fragment, getting the context of the activity required additional code that sometimes would be fine and, in some case, needed extra functions .

```
Volley.newRequestQueue(getActivity().getApplicationContext()).add(stringRequest);
```

Figure 61 - Fragment getActivity()

Also because of fragments, getting a calendar window to pop up for selecting a date was an incredible struggle. Eventually the application was able to load up the calendar window, but the data was lost because of the fragment it was in and requiring lots of additional code to pass out the data.

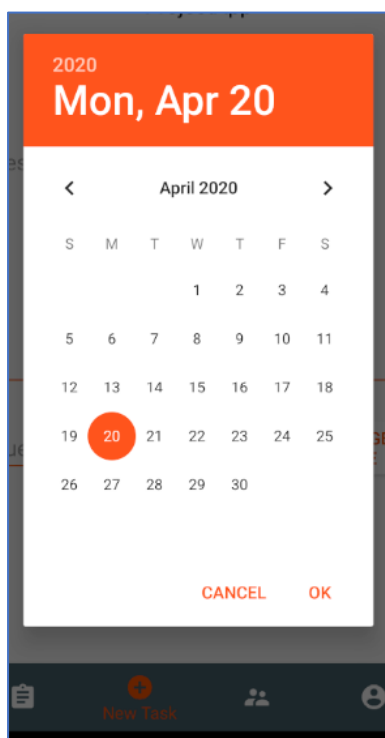


Figure 62 - Calendar window

Another bug that could not be figured was the navigation bar rising with the keyboard on only one page. Only the creating a task page would the navigation bar rise. After hours of studying the code, there was no indication to why it would rise. Eventually focus needed to be shifted from this to continue developing the main application.

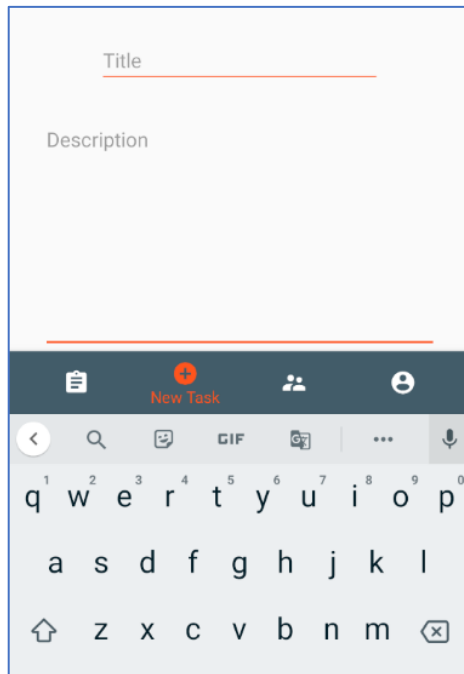


Figure 63 - Navigation bar bug

Because of lots of early development problems and slow learning of Java, the inclusion of user permissions needed to be excluded. The inclusion of notifications and camera access was a feature that looked simply when reaching the documentation, but development of the application needed to stop so the overall project did not suffer.

5.6.2 Database Problems

Most of the database problems came from implementing the task list. As started previously, the design of the code to include this feature needed to be separated from the rest of the functions, breaking the object orientated approach. The problem lied with getting the while loop to work. The file that help all the functions and executed them could not echo a JSON response. The problem consisted for close to a week with on and off development causing a lot of time to be lost. With the time lose, this is when the second solution of using it all in one file came about.

```

24     while($stmt->fetch()){
25         $temp = array();
26         $temp['id'] = $id;
27         $temp['TaskTitle'] = $TaskTitle;
28         $temp['TaskDescription'] = $TaskDescription;
29         $temp['TaskOwner'] = $TaskOwner;
30         $temp['TaskDate'] = $TaskDate;
31         $temp['TaskCompleted'] = $TaskCompleted;
32         array_push($tasks, $temp);
33     //}
34 }
35 echo json_encode($tasks);

```

Figure 64 - Problematic while loop

Using Postman to test the php files made the general debugging process for the application a breeze. All the information I needed was always displayed and error messages were easy to look up because Postman can display php error messages.

6 Testing

Throughout the entire development of the application, testing was done when ever a new feature was implemented into the application. This helped to make sure all the new code being added to the project worked correctly. Once a date to finish working on the application was set, the application was tested as if it was in an environment filled with different teachers and students. This theoretical use case helped to fine any last-minute bugs that were hurting the user experience. The process involved creating multiple teacher and students accounts into a blank database. Then different tasks were created from different teacher account to find if they linked properly over many students.

However, one major flaw found while testing was completing a task. How the database was structured only allowed the completion of a task from one source. An example would be a teacher setting a task to multiple students. When one of the students completed the task, the task would then be marked as completed for the teacher and all other students. This problem was an oversight while developing and would need to take a lot of additional work to fix.

6.1 User Testing Environment

As mentioned previously, being able to test on a physical device was not an option. By not owning an Android device, the program was unable to be installed on to a mobile device. This is by far the most ideal circumstances and would have liked to be to hand over a device that most people would be used to for testing purposes. However, the laptop owned has a touchscreen that folds backwards to act like a table device (Lenovo, inc., 2020). This was used in replacement to a mobile device. The laptop was orientated vertically to act like a smartphone, giving the best possible testing environment. The user could then use the touchscreens to work the app like they would a smart phone.

Before handing the application to testers, testing emails will be created to log in with. Then a brief explanation was given on what the application could be used for. Once the tester had been informed of the applications design concept, the tester would be free to explore the app and attempt to

create several users and tasks. The purpose of this testing methodology was to see if the application was designed simply enough so the users could navigate and work the app with ease.

After testing had finished, a series of questions were asked to the tester.

- Do you think design of the application is clear and easy to understand?
- Are there any changes you would make to the design to improve the assessability of the application?
- Do you feel this application would be useful for you?
 - If so, why?

6.2 User Feedback

The overall feedback received was positive. Testers felt the application was easy use and navigate around. The icons used throughout the application were familiar and did what they expected. One tester said that the app was very simple, but in a good way. The simplicity meant that a student would be able to pick it up and have learn it within minutes. Only having the two buttons for a student made the design clear and easy to read. The task list was clear and easy to understand, tasks showed all the information they needed without cluttering the screen. The tester liked how simple it was to tick off tasks and the clear check mark that was displayed when it was checked off.

However, there were some negative feedback. Testers felt it was unclear when a new account was made and were frustrated when met with a message saying the user email already existed. This could be fixed by increasing the time that the account creation message appeared on screen. Also, the tester liked the student view having the titles of the pages on the navigation bar and found it odd they were not present on the teacher view.

The tester felt that the application would be useful for them and could see themselves using a system like to set homework and or tasks to the students.

6.2.1 Suggested Changes

Feedback was received to be able to change the image shown on the task list page. They felt by being able to change this image to some that could represent a specific subject, would greatly enhance the experience for the students. The ability to add more customisation to the look of the application was a stretch goal that was originally thought of but was scrapped as it was too far out of scope for the project.

7. Critical Reflection

7.1 Project Planning

I was excited to start the project, and I feel the planning in the beginning of development suffered because of it. I had wanted to work on the idea in the background of my studies beforehand and had a lot of ideas being thrown about. Being able to narrow down the ideas thanks to the help of my supervisor, I was then able to plan appropriately for the rest of the development.

Sometimes when working on the deliverable, I would wonder from my original goal. I would either struggle to finish off a section or struggle to start a new task. this happened a lot in the beginning of development as I was overwhelmed by how much I needed to learn for one module. However, coming to the end of the development stage I felt comfortable with what I was developing and

started to finish off tasks as I was working on them. Also, back tracking and polishing up older code that I might have dropped as soon as it was at a working state.

7.2 Future Development

For the future development of the application, I want to be able to implement the feature that I did not manage to include into the project.

These features include:

- User permissions
- Push notifications
- Camera access
- File upload

If I can include these 4 features into the application in future development, the application would be a much more robust prototype with more features to entrench what the application's design is about. Being able to manage and track work for a wide range of students. To be able to implement these features I would need to research further into user permissions and the privacy policies that come along with them. Including these policies would mean I could develop the application to give the user more control over their work tracking.

If the application were to be handed to another developer, I feel that the naming convention I have used and put into place would be enough for work to continue.

7.3 Conclusion

Overall, I am happy with what I have been able to achieve within this project. I know I could have done more with the deliverable. I wanted to include many more features and polish up the design of the application a lot more. Being able to include features such as file upload, push notifications and camera access, would have given me much more of an insight to app development to then be able to continue working on the application in the future. Being able to connect my database to the application was the biggest accomplishment of the project. Connecting them together and adding more and more functions was satisfying and enjoyable. Once I had a good understanding of both PHP and Java, I was able to design quickly additional parts to the app. The speed I was working towards the end of the project gave me the confidence to finish off the tasks that I needed to do.

The research I managed to find on how applications are developed to cater to all ages and types of people was very insightful. The research has opened my eyes on the level of detail that developers put in to making their applications. Now knowing this I can apply this knowledge to other aspects of my future work when trying to accommodate all ages and types of people. Along with gaining knowledge to accommodate everyone when developing software, I have gained the general knowledge to be able to research thoroughly into a topic and find examples and resources related to the topic in question.

I have managed to create a prototype application that can be used to benefit the teaching experience for a wide range of ages. With some future development I feel I will be able to continue working on the project to produce a fully working concept that I can be happy to show off to people.

8. References

- Center of Universal Design in North Carolina. (n.d.). *Universal Design Principles* . Retrieved from housing.gov.bc.ca/pub/htmldocs/pub_universaldesign/design.pdf
- Anchit. (2017). *Converting a String to MD5 Hashes in Android*. Retrieved from mobikul: <https://mobikul.com/converting-string-md5-hashes-android/>
- Apache Friends, inc. . (2020). *XAMPP Apache + MariaDB + PHP + Perl*. Retrieved from Apache Friends: <https://www.apachefriends.org/index.html>
- Apple, Inc. (2016, December 8). *Start Developing iOS Apps (Swift)*. Retrieved from Developer.Apple: <https://developer.apple.com/library/archive/referencelibrary/GettingStarted/DevelopiOSAppsSwift/>
- Apple, Inc. (2020). *App Store Preview*. Retrieved from App Store: <https://apps.apple.com/gb/app/twitter/id333903271>
- Apple, inc. (2020). *Tab Bars*. Retrieved from developer.apple: <https://developer.apple.com/design/human-interface-guidelines/ios/bars/tab-bars/>
- Babich, N. (2018, March 27). *Animated Transitions in Mobile Apps*. Retrieved from UX Planet: <https://uxplanet.org/animated-transitions-in-mobile-apps-412b8e8478e7>
- Bennetch, I. (20202). *About*. Retrieved from phpmyadmin: phpMyAdmin
- Blackboard, inc. (2017, January 27). *Blackboard Delivers Worldwide Growth*. Retrieved from BlackBoard NewsRoom: <https://press.blackboard.com/2017-01-27-Blackboard-Delivers-Worldwide-Growth>
- Blair, I. (2020). *21 Design Tips For Award Winning Apps*. Retrieved from BuildFire: <https://buildfire.com/best-design-tips-mobile-apps/>
- Clyde, O. (2019, October 28). *What Is Trello and Why Should You Use It in Your Business?* Retrieved from CloudApp: <https://www.getcloudapp.com/blog/what-is-trello>
- Department for Education. (2019, October 11). *GOV.UK*. Retrieved from School funding allocations 2020-21: <https://dfemedia.blog.gov.uk/2019/10/11/school-funding-allocations-2020-21/>
- DiBiasio, R. (2020, January 21). *User Review of Blackboard* . Retrieved from TrustRadius: <https://www.trustradius.com/reviews/blackboard-learn-2020-01-18-19-06-35>
- Extensible Markup Language (XML) 1.0 (Fifth Edition)*. (n.d.). Retrieved from <https://www.w3.org/TR/REC-xml/>
- firefly, inc. (2020). *Online homework management system for schools*. Retrieved from Fireflylearning: <https://fireflylearning.com/benefits-of-online-homework>
- Geerlings, C. (2019, October 21). *How to create an accessible app (and why you should)*. Retrieved from Medium: <https://medium.com/oberonamsterdam/how-to-create-an-accessible-app-and-why-you-should-5493f41f8bdb>
- Google, inc. (2019). *Adapter*. Retrieved from developer.android: <https://developer.android.com/reference/android/widget/Adapter>

Google, inc. (2019). *Volley Overview*. Retrieved from developer.android:
<https://developer.android.com/training/volley>

Google, inc. (2020). *Android 4.3 APIs*. Retrieved from developer.android:
<https://developer.android.com/about/versions/android-4.3.html>

Google, inc. (2020). *Android 5.0 APIs*. Retrieved from developer.android:
<https://developer.android.com/about/versions/android-5.0.html>

Google, inc. (2020). *Bottom Navigation*. Retrieved from Material Design:
<https://material.io/components/bottom-navigation/#usage>

Google, inc. (2020). *Fragments*. Retrieved from developer.android:
<https://developer.android.com/guide/components/fragments>

Google, inc. (2020). *Kotlin*. Retrieved from Developer.android: <https://developer.android.com/kotlin>

Google, inc. (2020). *Material Design*. Retrieved from Material Design: <https://material.io/>

Google, Inc. (n.d.). *Android Studio*. Retrieved from Developer.Android:
<https://developer.android.com/studio>

JetBrains, inc. (2020). *Kotlin*. Retrieved from Kotlin: <https://kotlinlang.org/>

Jonsdottir, A. H. (2017). Difference in Learning Among Students Doing Pen-and-Paper Homework Compared to Web-Based Homework in an Introductory Statistics Course. 20.

Komal, F. (2019, July 6). *Apple app store gained more popularity than Google play store in first half of 2019*. Retrieved from Digital Infomation World:
<https://www.digitalinformationworld.com/2019/07/app-store-more-popularity-than-play-store.html>

Langly, N. (2002, May 2). *Write once, run anywhere?* Retrieved from Computer Weekly:
<https://www.computerweekly.com/feature/Write-once-run-anywhere>

Lenovo, inc. (2020). *Yoga 530*. Retrieved from Lenovo:
<https://www.lenovo.com/gb/en/laptops/yoga/500-series/Yoga-530-14-Intel/p/88YG5000978>

Mackenzie, T. (2012, May 7). *App store fees, percentages, and payouts: What developers need to know*. Retrieved from techRepublic: <https://www.techrepublic.com/blog/software-engineer/app-store-fees-percentages-and-payouts-what-developers-need-to-know/>

Microsoft OneNote Ratings and Reviews. (2020). Retrieved from App Store:
<https://apps.apple.com/gb/app/microsoft-onenote/id410395246#see-all/reviews>

MIT Media Lab. (n.d.). Retrieved from Scratch: <https://scratch.mit.edu/>

Oracle, inc. (2020). *go Java*. Retrieved from Java.com: <https://go.java/?intcmp=gojava-banner-java-com>

Oracle, inc. (2020). *MySQL*. Retrieved from MySQL: <https://www.mysql.com/>

Postman, inc. (2020). *The Collaboration Platform for API Development*. Retrieved from Postman:
<https://www.postman.com/>

- RhodeCode, Inc. (2016). *Version Control Systems Popularity in 2016*. Retrieved from RhodeCode: <https://rhodecode.com/insights/version-control-systems-2016>
- Rouse, M. (2017). *Definition MD5*. Retrieved from Search Security: <https://searchsecurity.techtarget.com/definition/MD5>
- Satchel, inc. (2020). *Satchel one in Secondary*. Retrieved from teamsatchel: <https://www.teamsatchel.com/schools/secondary.html>
- Satchel, inc. (2020). *Welcom to Satchel*. Retrieved from teamsatchel.com: <https://www.teamsatchel.com/>
- Slank, inc. (2020). *What is the best alternative to Android Studio?* Retrieved from Slant: <https://www.slant.co/options/4368/alternatives/~android-studio-alternatives>
- Square, Inc. (2020). *Retrofit*. Retrieved from Retrofit: <https://square.github.io/retrofit/>
- StatCounter, inc. (2020, March). *Android version market share worldwide*. Retrieved from Statcounter: <https://gs.statcounter.com/os-version-market-share/android/mobile-tablet/worldwide>
- StatCounter, inc. (2020, March). *Desktop vs Mobile vs Tablet Market Share Worldwide*. Retrieved from Statcounter: <https://gs.statcounter.com/platform-market-share/desktop-mobile-tablet>
- StatCounter, inc. (2020, March). *iOS version market share worldwide*. Retrieved from StatCounter: <https://gs.statcounter.com/ios-version-market-share/mobile-tablet/worldwide>
- Statista, inc. (2020, Febuary 14). *Household internet penetration in the United Kingdom (UK) 1998-2019*. Retrieved from Statista: <https://www.statista.com/statistics/275999/household-internet-penetration-in-great-britain/>
- Summerfield, J. (n.d.). *For Broad Marketing Outreach, A Mobile Website is the Place to Start*. Retrieved from Human Servies Solutions: <https://www.hswsolutions.com/services/mobile-web-development/mobile-website-vs-apps/>
- The Apache Software Foundation, inc. (2020). *Apache*. Retrieved from Apache: <https://www.apache.org/>
- the PHP Group. (2020). *What is PHP*. Retrieved from php: <https://www.php.net/manual/en/intro-what-is.php>
- Trussell, J. (2020). Comparison of the Effectiveness of Online. *IEEE TRANSACTIONS ON EDUCATION*, 7.
- Walmsley, M. (n.d.). Retrieved from Code Avengers: <https://www.codeavengers.com/>
- Weale, S. (2019, December Thursday). *More than 28% of England's secondary schools now in the red, study finds*. Retrieved from The Guardian: 19
- Willson, G. (2018, September 25). *Accessibility in Disability: Revolutionizing Mobile Apps*. Retrieved from Apptentive: <https://www.apptentive.com/blog/2018/09/25/accessibility-in-disability-revolutionizing-mobile-apps/>

PROJECT SPECIFICATION - Project (Technical Computing) 2019/20

| | |
|--------------------------|---|
| Student: | Jonathan Cunnew |
| Date: | 23/09/2019 |
| Supervisor: | Michael Bass |
| Degree Course: | Computer Science for Games |
| Title of Project: | Creating and easy to use work tracker for staff and students |

Elaboration

Tracking the amount of work that a student has done is difficult. It can be difficult for teachers to find the time to make sure everyone is keeping up to date with the work that is set. This project aims to build a tool that is focused around helping teachers and their students to manage and track the amount of work being done.

I am going to try and tackle this problem by developing a user-friendly application for staff and students where the students can check off completed work tasks and have the staff see any completed or incomplete work.

The prototype will be an Android application that can connect the staff and their students together via individual logins. The staff will be able to send tasks and see if those tasks have been viewed and even completed. The students will be able to access the work via the application, read feedback given and comment any questions. Each time a staff member assigns a piece of work the students will be notified and can view the work directly in the app. Some examples of work that could be send over the app would be in app question and answer forms, links to downloadable documents or webpages.

The core functionality I am aiming for is to allow staff be able to send work to their assigned students and have the students be able to check off in the app if they have completed the work and comment if they had any problems.

Project Aims

- Design a very easy to use user interface through research of a wide range of age groups to make the experience as easy as possible for all ages.
- Learn how to develop a simple app with Java and Android
- Research into user privacy and security for the log in system
- Be able to have a working system of sending projects to the linked student accounts
- Have a visual feedback for the staff when a student marks off work as completed
- Include options for the students when checking off projects. Example: completed, struggled too easy.

Project deliverable(s)

I will develop a prototype of an Android application for smart phones and tablets that is capable of

- Creating personal accounts for staff and students
- Having the student accounts linked to the appropriate staff account
- Be able to send messages and projects to the students from a staff account
- Have a working check off button for each project on the student's accounts

The application will be made using the Android Studio integrated development environment (IDE) and written in Java since this is the most recommended for android development.

I plan on using Trello to organise and manage any notes and tasks that need to be completed.

I will also stick to a fortnightly sprint schedule to the workflow going while also being able to track my progress efficiently.

Action plan

| ACTION | DUE DATE |
|---|--------------------------------|
| FINDING A SUPERVISOR | 11 th October 2019 |
| PROJECT SPECIFICATION AND ETHICS FORM | 25 th October 2019 |
| RESEARCH <ul style="list-style-type: none"> • MAKING A SIMPLE ANDROID APP • LOG IN AND ACCOUNT SYSTEMS | 1 st November 2019 |
| SET UP VERSION CONTROL FOR A TESTING PROJECT (INITIAL SET UP OF AN APP) | 1st November 2019 |
| SET UP A TRELLO BOARD WITH A FORTNIGHTLY SPRINT SCHEDULE | 1 st November 2019 |
| CREATE A VERY SIMPLE APP RUNNING IN AN ANDROID EMULATION WITH AN ACCOUNT CREATION BUTTON | 22 nd November 2019 |
| RESEARCH <ul style="list-style-type: none"> • LOOK INTO ACCOUNT SAFETY AND PRIVACY • USER INTERFACE DESIGN CONCEPTS • FURTHER RESEARCH APP DEVELOPMENT FOR FEATURES INCLUDED IN THE PROJECT (TASK ASSIGNING BETWEEN STAFF AND STUDENT) | 4 th December 2019 |
| INFORMATION REVIEW | 6 th December 2019 |
| HAVE A WORKING PROTOTYPE OF MY APPLICATION THAT CAN CREATE ACCOUNTS, SEND TASK AND COMPLETE TASKS | 10 th January 2020 |
| WORKING PROTOTYPE WITH AN EASY TO USE UI IMPLEMENTED INTO THE DESIGN | 24 th January 2020 |

| | |
|--|--------------------------------|
| COMPILE A SURVEY ABOUT MY UI AND ACT ACCORDINGLY TO THE FEEDBACK | 5 th February 2020 |
| PROVISIONAL CONTENTS PAGE | 21 st February 2020 |
| DRAFT CRITICAL EVALUATION | 27 th March 2020 |
| SECTIONS OF DRAFT REPORT | 27 th March 2020 |
| SUBMIT TO TURNITIN | 22 nd April 2020 |

BCS Code of Conduct

I confirm that I have successfully completed the BCS code of conduct on-line test with a mark of 70% or above. This is a condition of completing the Project (Technical Computing) module.

Signature:



Publication of Work

I confirm that I understand the "Guidance on Publication Procedures" as described on the Bb site for the module.

Signature:



GDPR

I confirm that I will use the "Participant Information Sheet" as a basis for any survey, questionnaire or participant testing materials. This form is available on the Bb site for the module.

Signature:



Ethics

Complete the SHUREC 7 (research ethics checklist for students) form below. If you think that your project may include ethical issues that need resolving (working with vulnerable people, testing procedures etc.) then discuss this with your supervisor as soon as possible and comment further here.

Both you and your supervisor need to sign the completed SHUREC 7 form.

Please contact the project co-ordinator if further advice is needed.

Appendix B – Ethics Form

RESEARCH ETHICS CHECKLIST FOR STUDENTS (SHUREC 7)

This form is designed to help students and their supervisors to complete an ethical scrutiny of proposed research. The SHU [Research Ethics Policy](#) should be consulted before completing the form.

Answering the questions below will help you decide whether your proposed research requires ethical review by a Designated Research Ethics Working Group.

The final responsibility for ensuring that ethical research practices are followed rests with the supervisor for student research.

Note that students and staff are responsible for making suitable arrangements for keeping data secure and, if relevant, for keeping the identity of participants anonymous. They are also responsible for following SHU guidelines about data encryption and research data management.

The form also enables the University and Faculty to keep a record confirming that research conducted has been subjected to ethical scrutiny.

For student projects, the form may be completed by the student and the supervisor and/or module leader (as applicable). In all cases, it should be counter-signed by the supervisor and/or module leader and kept as a record showing that ethical scrutiny has occurred. Students should retain a copy for inclusion in their research projects, and staff should keep a copy in the student file.

Please note if it may be necessary to conduct a health and safety risk assessment for the proposed research. Further information can be obtained from the Faculty Safety Co-Ordinator.

General Details

| | |
|---|--|
| Name of student | Jonathan Cunnew |
| SHU email address | B6027956@my.shu.ac.uk |
| Course or qualification (student) | Computer Science for Games |
| Name of supervisor | Michael Bass |
| email address | m.bass@shu.ac.uk |
| Title of proposed research | Creating and easy to use work tracker for staff and students |
| Proposed start date | 23 rd September 2019 |
| Proposed end date | 22 nd April 2020 |
| Brief outline of research to include, rationale & aims (250-500 words). | My research will include looking at already existing work planners and finding features that they lack or excel very well in and reviewing what makes people want to use these features. in doing this I should be able to get a good understanding of what is missing in the market and I can then target further research towards solving these gaps in the market with my application. |

| | |
|--|--|
| | <p>A large amount of my research will be on what makes a well-designed user interface and ease of use for the consumer. By researching into this I will gain the required skills to be able to make the application simple and easy to use.</p> <p>The technical research will consist of me learning and understanding the Android Studio (IDE) and the Java coding language. Researching and learning both will enable me to develop a prototype application with the features to connect the staff and students.</p> <p>My initial target audience was ages 8 – 16 but the initial prototype will be developed in mind for all ages and any testing that may need doing will involve over 18's only.</p> |
| Where data is collected from individuals, outline the nature of data, details of anonymization, storage and disposal procedures if required (250-500 words). | <p>Once I can draft up a basic user interface will conduct a face to face questionnaire with willing participants from learning based backgrounds.</p> <p>I will have my application loaded on to a test android device and will hand it over to the tester. After a few minutes of use I will ask a series of questions, such as: what stood out for them, did they like the user interface, what felt lack luster.</p> <p>Each face to face questioning I will fill out a feedback form from their responses use this to improve the overall experience of my app.</p> <p>The feedback forms will be stored only on my secure student google drive and will be deleted once the project has come to a close.</p> |

1. Health Related Research Involving the NHS or Social Care / Community Care or the Criminal Justice Service or with research participants unable to provide informed consent

| Question | Yes/No |
|---|--------|
| <p>1. Does the research involve?</p> <ul style="list-style-type: none"> • Patients recruited because of their past or present use of the NHS or Social Care • Relatives/careers of patients recruited because of their past or present use of the NHS or Social Care • Access to data, organs or other bodily material of past or present NHS patients • Fetal material and IVF involving NHS patients • The recently dead in NHS premises • Prisoners or others within the criminal justice system recruited for health-related research* • Police, court officials, prisoners or others within the criminal justice system* • Participants who are unable to provide informed consent due to their incapacity even if the project is not health related | No |

| | |
|--|----|
| 2. Is this a research project as opposed to service evaluation or audit? | No |
| <i>For NHS definitions please see the following website</i> http://www.hra.nhs.uk/documents/2013/09/defining-research.pdf | |

If you have answered **YES** to questions **1 & 2** then you **must** seek the appropriate external approvals from the NHS, Social Care or the National Offender Management Service (NOMS) under their independent Research Governance schemes. Further information is provided below.

NHS <https://www.myresearchproject.org.uk/Signin.aspx>

* All prison projects also need National Offender Management Service (NOMS) Approval and Governor's Approval and may need Ministry of Justice approval. Further guidance at: <http://www.hra.nhs.uk/research-community/applying-for-approvals/national-offender-management-service-noms/>

NB FRECs provide Independent Scientific Review for NHS or SC research and initial scrutiny for ethics applications as required for university sponsorship of the research. Applicants can use the NHS pro-forma and submit this initially to their FREC.

2. Research with Human Participants

| Question | Yes/No |
|--|--------|
| Does the research involve human participants? This includes surveys, questionnaires, observing behavior etc. | Yes |
| Question | Yes/No |
| 1. <i>Note If YES, then please answer questions 2 to 10</i> <i>If NO, please go to Section 3</i> | |
| 2. Will any of the participants be vulnerable? <i>Note: Vulnerable' people include children and young people, people with learning disabilities, people who may be limited by age or sickness, etc. See definition on website</i> | No |
| 3. Are drugs, placebos or other substances (e.g. food substances, vitamins) to be administered to the study participants or will the study involve invasive, intrusive or potentially harmful procedures of any kind? | No |
| 4. Will tissue samples (including blood) be obtained from participants? | No |
| 5. Is pain or more than mild discomfort likely to result from the study? | No |
| 6. Will the study involve prolonged or repetitive testing? | No |
| 7. Is there any reasonable and foreseeable risk of physical or emotional harm to any of the participants? <i>Note: Harm may be caused by distressing or intrusive interview questions, uncomfortable procedures involving the participant, invasion of privacy, topics relating to highly personal information, topics relating to illegal activity, etc.</i> | No |
| 8. Will anyone be taking part without giving their informed consent? | No |
| 9. Is its covert research? <i>Note: 'Covert research' refers to research that is conducted without the knowledge of participants.</i> | No |

| | |
|---|----|
| 10. Will the research output allow identification of any individual who has not given their express consent to be identified? | No |
|---|----|

If you answered **YES only** to question 1, the checklist should be saved and any course procedures for submission followed. If you have answered **YES** to any of the other questions you are **required** to submit a SHUREC8A (or 8B) to the FREC. If you answered **YES** to question 8 and participants cannot provide informed consent due to their incapacity you must obtain the appropriate approvals from the NHS research governance system. Your supervisor will advise.

3. Research in Organizations

| Question | Yes/No |
|---|--------|
| 1. Will the research involve working with/within an organization (e.g. school, business, charity, museum, government department, international agency, etc.)? | |
| 2. If you answered YES to question 1, do you have granted access to conduct the research? <i>If YES, students please show evidence to your supervisor. PI should retain safely.</i> | |
| 3. If you answered NO to question 2, is it because: A. you have not yet asked B. you have asked and not yet received an answer C. you have asked and been refused access. <i>Note: You will only be able to start the research when you have been granted access.</i> | |

4. Research with Products and Artefacts

| Question | Yes/No |
|---|--------|
| 1. Will the research involve working with copyrighted documents, films, broadcasts, photographs, artworks, designs, products, programs, databases, networks, processes, existing datasets or secure data? | Yes |
| 2. If you answered YES to question 1, are the materials you intend to use in the public domain? <i>Notes: 'In the public domain' does not mean the same thing as 'publicly accessible'.</i> <ul style="list-style-type: none"> Information which is 'in the public domain' is no longer protected by copyright (i.e. copyright has either expired or been waived) and can be used without permission. Information which is 'publicly accessible' (e.g. TV broadcasts, websites, artworks, newspapers) is available for anyone to consult/view. It is still protected by copyright even if there is no copyright notice. In UK law, copyright protection is automatic and does not require a copyright statement, although it is always good practice to provide one. It is necessary to check the terms and conditions of use to find out exactly how the material may be reused etc. <i>If you answered YES to question 1, be aware that you may need to consider other ethics codes. For example, when conducting Internet research, consult the code of the Association of Internet Researchers; for educational research, consult the Code of Ethics of the British Educational Research Association.</i> | Yes |

| | |
|--|-------|
| 3. If you answered NO to question 2, do you have explicit permission to use these materials as data? <i>If YES, please show evidence to your supervisor.</i> | |
| 4. If you answered NO to question 3, is it because: A. you have not yet asked permission B. you have asked and not yet received and answer C. you have asked and been refused access. <i>Note: You will only be able to start the research when you have been granted permission</i> | A/B/C |

Adherence to SHU policy and procedures

| | |
|--|------------------|
| Personal statement | |
| I can confirm that: | |
| <input type="checkbox"/> I have read the Sheffield Hallam University Research Ethics Policy and Procedures <input type="checkbox"/> I agree to abide by its principles. | |
| Student | |
| Name: Jonathan Cunnew | Date: 17/10/2019 |
| Signature:  | |
| Supervisor or another person giving ethical sign-off | |
| I can confirm that completion of this form has not identified the need for ethical approval by the FREC or an NHS, Social Care or other external REC. The research will not commence until any approvals required under Sections 3 & 4 have been received. | |
| Name: Michael Bass | Date: 18/10/2019 |
| Signature: M. Bass | |

Appendix C – Abbreviations and Definitions/Glossary

Android - An open source mobile operating system primarily designed for touchscreen mobile devices such as smart phones.

Variable – A value to hold information and can be changed depending on the conditions of the information passed to it.

Function – A section of code that has been named so the coder can call a large amount of code without needed to rewrite all of it.

JSON – A format users for transmitting structured data over a network connection. This format is used primarily in web-based software's.

UI - User Interface

The look and design of a soft ward that the user physically interacts with.

IDE – Integrated Development Environment

A software application that provides comprehensive facilities to computer programmers for software development.

XML – Extensible Markup Language

A programming language that defines a set of rules for encoding documents. The data is stored in a human-readable and machine-readable format.

PHP – Personal Home Page Tools

A widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML (the PHP Group, 2020).

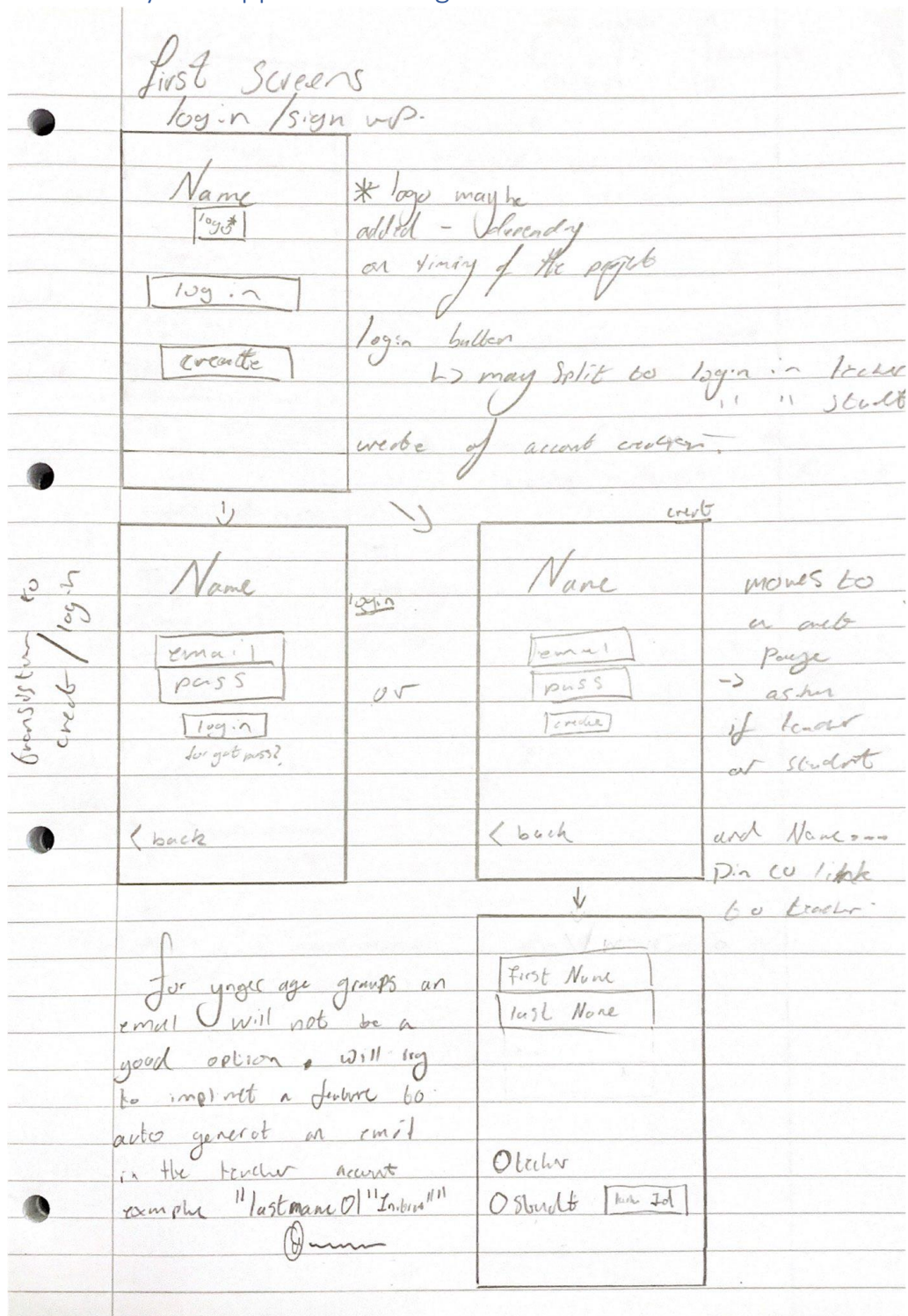
Java – A general purpose programming language designed to have a few dependencies as possible so code and work anywhere and everywhere.

Kotlin – A cross-platform programming language designed to interoperate Java. Along with creating a user-friendly programming experience.

VLE - Virtual Learning Environment

A teaching and learning tool designed to enhance a students' learning experience by including computers and the internet within the learning process.

Appendix D – Physical Application Designs



teacher view

full width buttons

Scrollable

Icons used as a sort of archive needing a picture.

Icons

Name

Students

Sign out

Settings

Logout

List + Ac

Ac - account

- gives access to account and any connected students

+ - create a new 'task'

List - See all of set work

- See all students profiles.

blank space fills with the option selected - default to Account

Create Task view (+)

Title -----

Description -----

File upload

Date select

date drop down box

creating a task with name

adding a ; title -

; description - Max 2000 chars

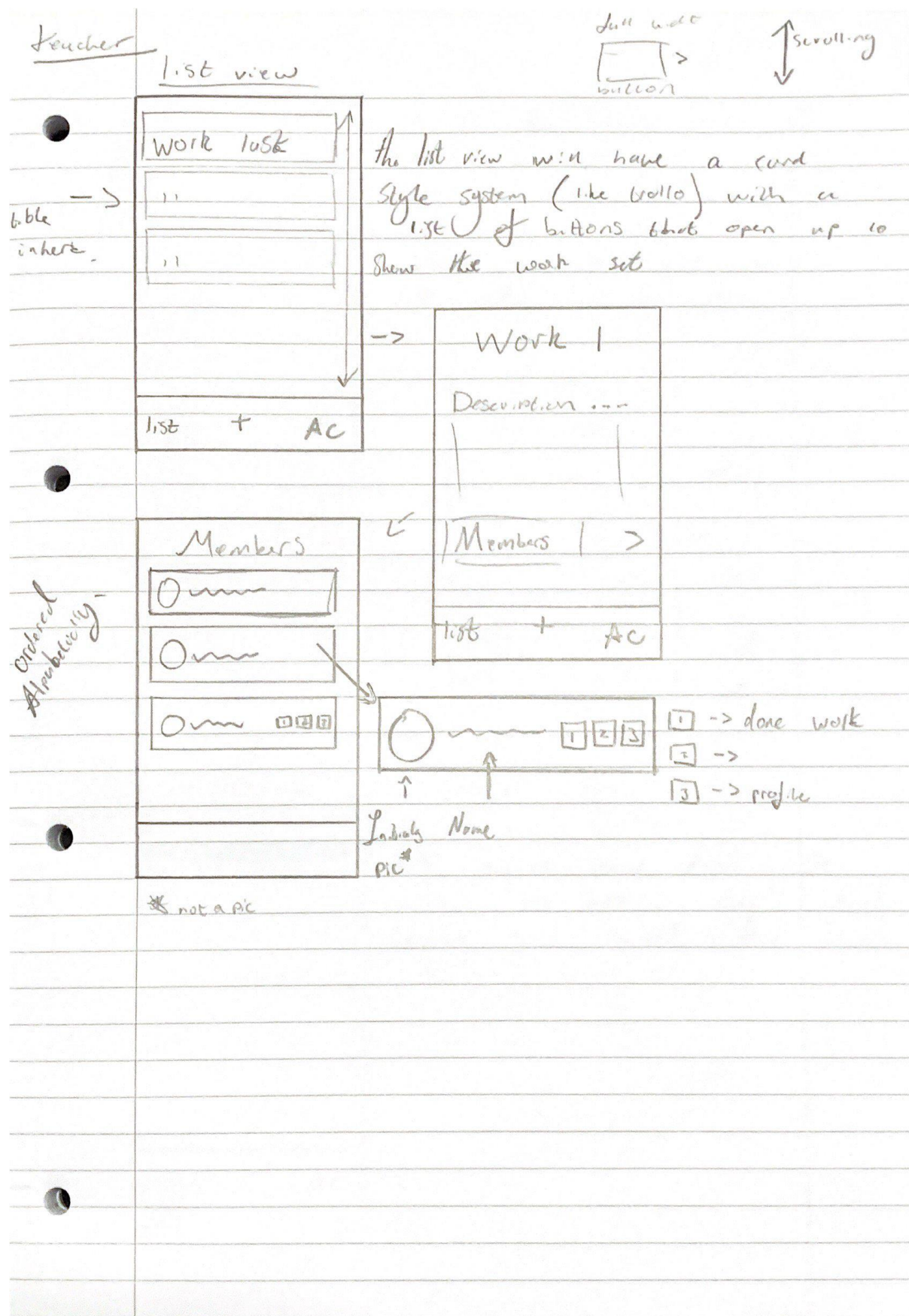
; File upload - not mandatory

- only Doc.

; date

List + Ac

bottom guide access menu is always available for quick easy navigation.



child

↑ scrolling
↓

or

| | |
|-------------|---|
| Title | > |
| Title | > |
| Due date | |
| Attachment? | |

| | |
|------|----|
| list | AC |
|------|----|

unlike the teacher view ~~A~~
child account will default
on to the 'list' view.

Opening up right on to
the tasks set for them

→

| | |
|-----------------|----|
| Title | ↑ |
| (due date) | |
| Description ... | |
| Attachment | ↓ |
| list | AC |

opens to
an extended
view of the
task

Setting set
Title Description
Attachment
and due
date

Noben and other on screen intrusions

| | |
|-----------------------|------|
| XXXXXXXXXX | |
| | |
| | |
| | |
| | |
| list | + AC |

~~XXXX~~ - blank space filled with
colour to avoid any information
getting lost behind the noben.

Appendix E – larger screenshots

```
//asking for the strings to send them to the database
StringRequest stringRequest = new StringRequest(Request.Method.POST,
    php_Constants.URL_REGISTER,
    new Response.Listener<String>() {
        @Override
        public void onResponse(String response) {
            progress.dismiss();
            try {
                JSONObject jsonObject = new JSONObject(response);
                Toast.makeText(getActivity().getApplicationContext(), jsonObject.getString("message"), Toast.LENGTH_LONG).show();
                //response boxes are not showing due to a fix to get past HTTP connection errors
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }
    },
    new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            progress.hide();
            Toast.makeText(getActivity().getApplicationContext(), error.getMessage(), Toast.LENGTH_SHORT).show();
        }
    }) {
    @Override
    protected Map<String, String> getParams() throws AuthFailureError {
        Map<String, String> params = new HashMap<>();
        params.put("firstname", firstNameSr);
        params.put("lastname", lastNameSr);
        params.put("email", emailSr);
        params.put("password", passwordSr);
        params.put("isStudent", isStudentSr);
        params.put("accOwner", accOwnerSr);
        return params;
    }
};
RequestHandlerSingleton.getInstance(getActivity().getApplicationContext()).addToRequestQueue(stringRequest);
}
```

Appendix E 1 - Registering a User Android String Request

```
WebServices > GetUserTasks.php > ...
1  <?php
2
3      //this is the one being used in side the app
4      //trying to impliment this while loop inside the oop format of the other operations was not going well
5      //resorted to including it all in 1 file here
6      define('DB_NAME','fyp');
7      define('DB_USER','root');
8      define('DB_PASSWORD','');
9      define('DB_HOST','localhost');
10     $con = new mysqli(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);
11     if(mysqli_connect_errno()){
12         die('Unable able to connect '. mysqli_connect_error());
13     }
14
15     // if($_SERVER['REQUEST_METHOD']=='POST'){
16     // $taskowner = $_POST['taskowner'];
17     $stmt = $con->prepare("SELECT* FROM tasks");// WHERE TaskOwner = ?");
18     // $stmt->bind_param("s",$taskowner);
19     $stmt->execute();
20     $stmt->bind_result($id, $TaskTitle,$TaskDescription,$TaskOwner,$TaskDate,$TaskCompleted);
21     $tasks = array();
22
23     while($stmt->fetch()){
24         $temp = array();
25         $temp['id'] = $id;
26         $temp['TaskTitle'] = $TaskTitle;
27         $temp['TaskDescription'] = $TaskDescription;
28         $temp['TaskOwner'] = $TaskOwner;
29         $temp['TaskDate'] = $TaskDate;
30         $temp['TaskCompleted'] = $TaskCompleted;
31         array_push($tasks, $temp);
32     }
33 }
34 echo json_encode($tasks);
```

Appendix E 2 - Full php script for collecting all tasks

```

public void setOnItemClickListener(OnItemClickListener listener) { mListener = listener; }

public TaskAdapter(Context mContext, List<TasksManager> tasksManagerList) {
    this.mContext = mContext;
    this.tasksManagerList = tasksManagerList;
}

@NonNull
@Override
public taskViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
    LayoutInflater inflater = LayoutInflater.from(mContext);
    View view = inflater.inflate(R.layout.task_card_layout, root: null);
    return new taskViewHolder(view);
}

@Override
public void onBindViewHolder(@NonNull taskViewHolder holder, int position) {
    TasksManager tasksManager = tasksManagerList.get(position);
    holder.cardTitle.setText(tasksManager.getTaskTitle());
    holder.cardOwner.setText(tasksManager.getTaskOwner());
    holder.cardDate.setText(tasksManager.getTaskDate());

    if(tasksManager.getTaskComp().equals("false")){
        holder.cardCompleted.setChecked(false);
    }else if(tasksManager.getTaskComp().equals("true")) {
        holder.cardCompleted.setChecked(true);
    }
}

@Override
public int getItemCount() {
    return tasksManagerList.size();
}

class taskViewHolder extends RecyclerView.ViewHolder{

    TextView cardTitle,cardOwner;
    TextView cardDate;
    CheckBox cardCompleted;
    ImageView cardIFAttachment;
    ImageView cardImage;
    ConstraintLayout taskLayout;

    public taskViewHolder(@NonNull View itemView) {
        super(itemView);
        cardTitle = itemView.findViewById(R.id.taskTitle);
        cardOwner = itemView.findViewById(R.id.taskOwned);
        cardCompleted = itemView.findViewById(R.id.taskCompletedBox);
        cardDate = itemView.findViewById(R.id.taskDate);
        cardIFAttachment = itemView.findViewById(R.id.taskAttachmentIncluded);
        cardImage = itemView.findViewById(R.id.taskImage);
        taskLayout = itemView.findViewById(R.id.layoutConstraint);

        itemView.setOnClickListener((v) -> {
            if(mListener != null){
                int pos = getAdapterPosition();
                if(pos != RecyclerView.NO_POSITION){
                    mListener.OnItemClick(pos);
                }
            }
        });
    }
}

```

```

//https://mobikul.com/convert-string-md5-hashes-android/
//code take from ^^
//used to try and match old passwords together
//md5 in android was giving a very slightly different md5 hash
public String md5(String s) {
    try {
        // Create MD5 Hash
        MessageDigest digest = java.security.MessageDigest.getInstance("MD5");
        digest.update(s.getBytes());
        byte messageDigest[] = digest.digest();

        // Create Hex String
        StringBuffer hexString = new StringBuffer();
        for (int i=0; i<messageDigest.length; i++)
            hexString.append(Integer.toHexString(0xFF & messageDigest[i]));

        return hexString.toString();
    } catch (NoSuchAlgorithmException e) {
        e.printStackTrace();
    }
    return "";
}
}

```

Appendix E 4 - Attempt to use MD5 in Android

- (Anchit, 2017)