

Southern Institute of Technology

Assignment 1

IT617 Mobile Application Development

Mobile Application Project – Study Tracker Application

Paige Clay 2019012408

Date of Submission: 19th June, 2024

Table of Contents

Table of Contents.....	2
Introduction.....	4
Design.....	5
Wireframes.....	5
Home/Main Activity.....	5
Timetable Activity – With View Pager.....	5
To-Do List Activity	6
Add Fragment.....	6
Delete/Edit Fragment	7
Material Design	7
FAB(Floating Action Bar)	7
Colours	7
Buttons	7
Snackbar	8
Layout.....	8
Small vs Large Devices	8
Database.....	9
ERD(Entity Relationship Diagram).....	9
How the Application Interacts with the Database	9
Implementation.....	10
Activity – Home	10
Phone	10
Tablet.....	11
Activity – Timetable.....	11
Phone	12
Tablet.....	13
Activity – To-Do List	13
Phone	14
Tablet.....	14
Fragment – Add	15
Phone	15
Tablet.....	16
Fragment – Delete/Edit	16
Phone	17

Tablet.....	17
Fragment – Add Subject	17
Phone	18
Tablet.....	18
String Resources	19
RecyclerViews.....	19
Testing	20
Pixel Pro 7 8.0 vs Pixel 7 11.0	20
Pixel C 8.0 vs Pixel C 11.0.....	21
Phones vs Tablets	22
Conclusion	23
References.....	24

Introduction

According to coursera.org, “time management is the process of consciously planning and controlling time spent on specific tasks to increase how efficient you are.” Time management is such an important skill to have, whether you are a full-time high school student, a 40-year-old part-time Polytech learner or something in between.

When you are studying, it can be difficult to keep up with all the labs, assignments, and exams, especially when they are coming from several classes at once. They all have different due dates, urgencies and difficulties which can make it hard to decide which is the best thing to focus on.

This is why I have decided to create an app that will allow students to track tasks that they input into the application so they can see everything they need to do in one place. It will also have a timetable so that they can keep track of when they have their classes. This app will be developed using Android Studio and the languages XML, Java, and SQL(lite).

The target users will be students of age 10+, which are high school students and adult learners. This is because the majority of these students will have access to a mobile device.

My goal is for this app to increase the user’s ability to manage their time, which in turn will provide these benefits:

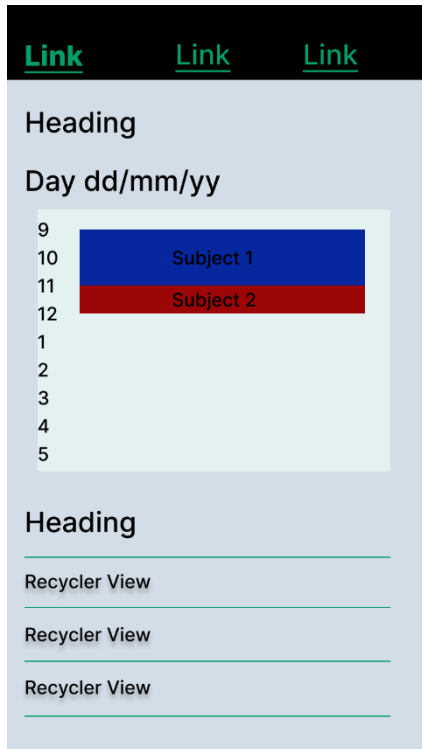
- Allow student to better allocate their time
- Clear goals and items to work on
- Reduce procrastination
- Increase efficiency and productivity
- Greater output of work

Design

Wireframes

Below are the wireframes for the phone layouts. Due to the minimal differences between the phone and tablet designs, tablet wireframes are not included.

Home/Main Activity



- In the home activity timetable periods for only the current day are displayed on a calendar-like grid.
- Tasks due for only the current date are shown in a list of RecyclerViews
- The app gets the local day and date to display the relevant timetable periods and tasks.
- Navigation works similarly to website links so tapping on a text link will navigate to the associated activity. The current activity's text link goes bold to indicate the user's location.

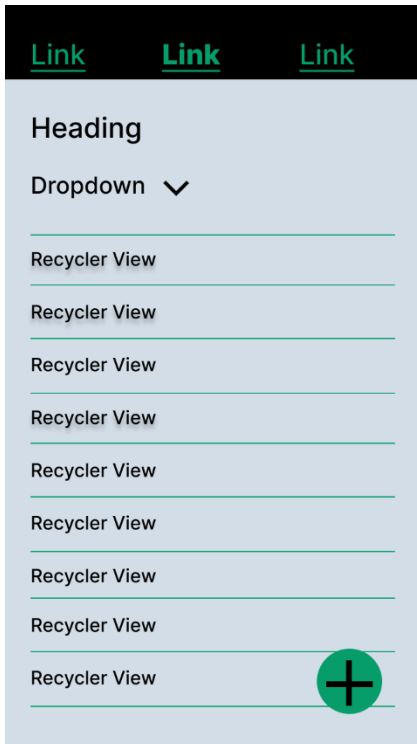
Timetable Activity – With View Pager



- The timetable activity hosts three fragments:
 - Daily timetable fragment which displays timetable periods like the home activity.
 - Weekly timetable fragment which displays timetable periods in a weekly calendar-like grid.

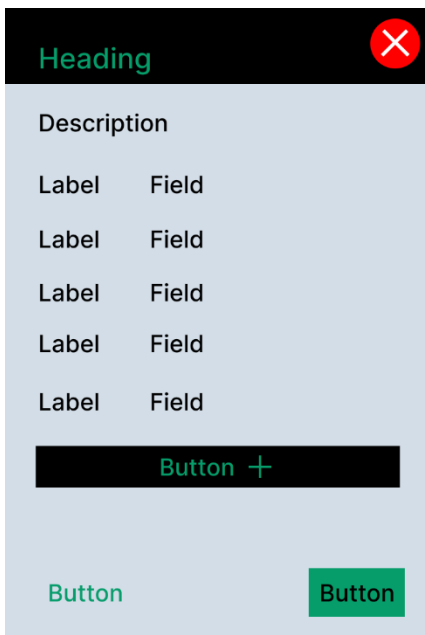
- By subject fragment which displays RecyclerViews grouped by subject, which are then sorted by the day and start time.
- The view pager and positioning determine which fragment to display. The first to display is the daily timetable fragment.
- Users swipe horizontally to switch fragments.
- The tab layout under the main navigation indicates the current fragment.
- The FAB(Floating Action Button) allows users to add a new timetable period by popping up a form fragment.

To-Do List Activity



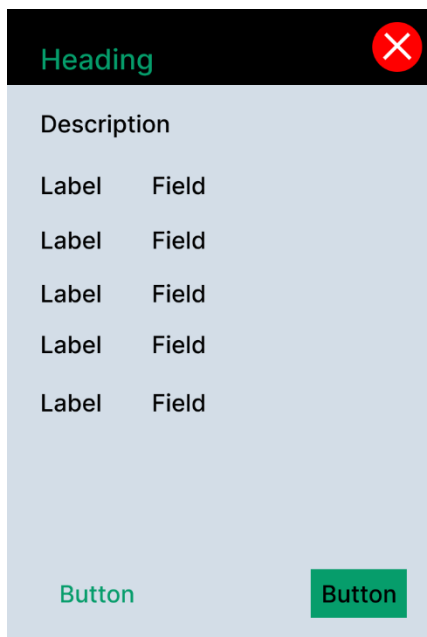
- The to-do list activity uses RecyclerViews to display all tasks in the database.
- A dropdown(spinner) allows users to choose the order of tasks, e.g. by due date, or by urgency. The default is by due date.
- The FAB(Floating Action Button) allows users to add a new task by popping up a form fragment.

Add Fragment



- The add fragment pops up when a FAB is tapped.
- It acts as a form for entering a new timetable period or task.
- Data entry fields include text fields, spinners, and date pickers.
- Buttons:
 - Red circle with cross closes the fragment without saving.
 - Black rectangle with plus adds a new section.
 - Text button deletes the last section.
 - Green button closes the fragment and sends the data to the database

Delete/Edit Fragment

A dialog box for deleting or editing a fragment. It has a black header with the word "Heading" in green and a red circle with a white 'X' in the top right corner. Below the header is a light blue area with the word "Description" in bold. Under "Description" is a table with five rows, each containing a "Label" and a "Field". At the bottom of the dialog are two buttons: a green one labeled "Button" and a black one labeled "Button".

Description	
Label	Field
Label	Field
Label	Field
Label	Field
Label	Field

- The delete/edit fragment pops up when a RecyclerView or period item is double-tapped.
- All fields are already populated with the item's data.
- Buttons:
 - Red circle with cross closes the fragment without saving.
 - Text button deletes the item from the database and closes the fragment.
 - Green button saves changes to the database and closes the fragment.

Material Design

Material Design 3 was created by Google to help developers follow best practices to ensure that the user's experience is enjoyable and positive. Below is how I am planning on implementing Material Design 3 into my app.

FAB(Floating Action Bar)

I have a FAB in the Timetable Activity, and one in the Tasks Activity. The one that is in the Timetable Activity will allow the user to add a new entry(class/period) to the Timetable table in the database. The one that is in the Tasks Activity will do the same except it will be for the Tasks table. Using a FAB in the activities will show the users that there is a primary action that can be done on the activity. Since the icon is a plus, it will illustrate clearly what the function is.



Colours

When designing this app, I kept in mind that bright colours can often be too stimulating and distracting, so I stayed away from using white and went for an off-white gray.



For the colour scheme of my app, I used colors.co as it has a useful colour scheme generator that allows me to lock in a colour that I like, and then generates a palette based off of that colour. This function helped me to decide on a basic three-coloured scheme that loosely follows an analogous colour scheme, as shamrock green(teal green) and azure(light blue gray) are adjacent on the colour wheel, but with different levels of light and dark for contrast, and black is serving as a neutral colour.

Buttons

In the fragments where the user can add a new entry to a database table there are three buttons, one will be "Clear", the other will be "Save/Add", and one will be "Add Section" so multiple entries can be added at once, it will be similar for the delete/edit fragments except there will be two buttons and one will be "Delete" and the other will be "Save". Since these buttons will have two very different functions, the user must be able to distinguish between them to avoid accidentally tapping on the wrong one.



To do this, each button has a distinctive design. The buttons that save the entry are filled buttons to make them stand out, the buttons that delete or clear the information are text buttons so that they have less emphasis, and the buttons that add a new section will be another filled button, but with different colours and an increased width.



Button



In addition to the main buttons on the fragments, there will also be a red icon with a cross in the middle to signify an exit or cancel button.

Snackbar

Snackbars are useful to display brief updates about processes occurring in the app or to warn the user when something isn't working.

I intend on using the snackbar to inform the user when they don't fill out the required fields in the form e.g. when they create a new timetable period and only input the subject name and start time, they push save the incomplete data will not be saved and a snackbar will appear saying "Please fill out all fields".

Layout

Small vs Large Devices

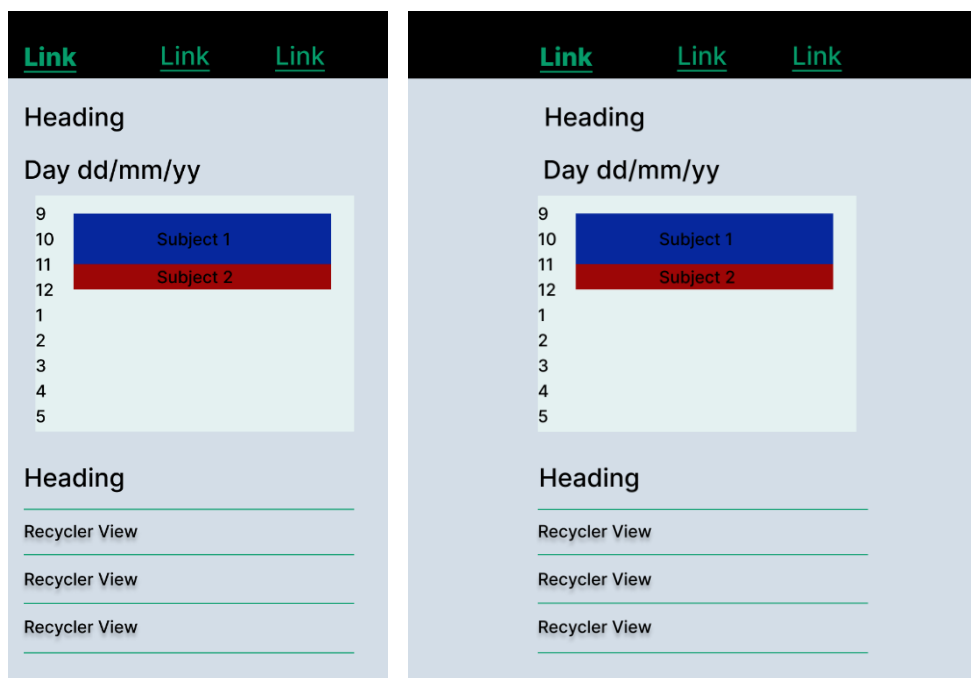
More regarding this will be discussed in greater detail later in the document so below is a basic summary of the differences between small and large devices

The layout of the app was first designed for a Pixel 7 Pro smartphone that supports Android 11. The font sizes are 30sp for headings/titles, 24 for subheadings, and 20sp for body. This is for visual hierarchy.

The other layout has been designed to work on a tablet that also supports Android 11. The font sizes are 40sp for headings/titles, 30 for subheadings, and 26sp for the body. They are scaled up to ensure effective use of the tablet-sized screen real estate. In addition to the increased font size, the FAB and the exit/cancel buttons have an increased size.

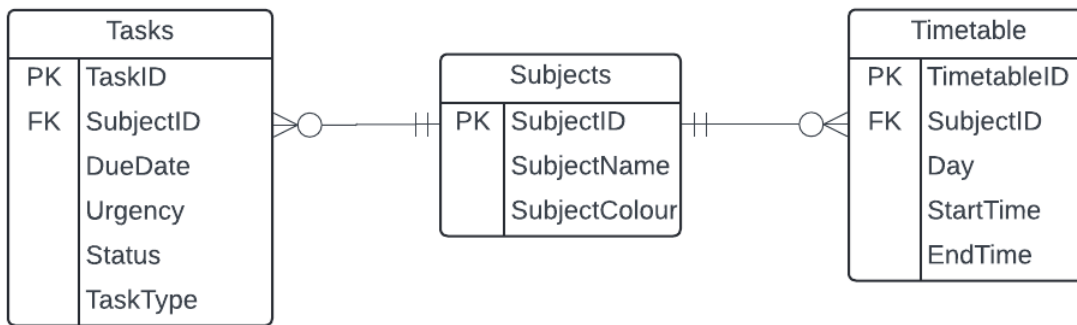


The main difference between the phone and tablet layout is the use of white space. As you can see from the example below, for the tablet layout the body has been placed in the horizontal centre and there are columns of empty space between them.



Database

ERD(Entity Relationship Diagram)



In this ERD diagram, it shows the three tables and the relationships between them. The Subject's table is the main entity as the other tables rely on its Primary Key(SubjectID). The line between the Tasks and Subjects tables shows that a task can only be assigned one and only one subject, and a subject can have zero or many assignments. This is similar to the relationship between the Subjects and Timetable tables because a class/period can be assigned one and only one subject, and a subject can have zero or many classes/periods.

How the Application Interacts with the Database

For adding information to the database, the user pushes the FAB, and it will pull up a fragment with labels and corresponding empty fields that acts as a form. The user can enter the data in the form and then save it which will add the data entered to the database. They can also push the Add Section button which allows the user to add multiple entries into the database without needing to click the FAB to bring up the form.

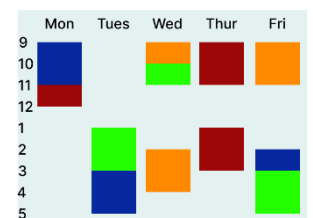
When editing or deleting a form, the user can tap on a RecyclerView, and the delete/edit fragment will pull up with data already populated in the form's fields. This allows the user to adjust any of the data and then save it. Or by pushing the delete button they can completely remove the entry from the database.

RecyclerView

The application mainly uses RecyclerViews to display the data from the database. This is because they are efficient for displaying large sets of data and unlike a list, can display multiple fields in a single RecyclerView instead of just one field. I will use adapters to implement them.

The other way I want to display the data, which is only for the classes/periods in the Timetable table, is in a calendar-like grid that uses the day, start time, and end time of the classes to figure out where on the grid the class-block will go. block will go.

These CRUD(create, read, update, and delete) methods to do the above are put in a special Java class called a DatabaseHelper, and each method has SQLite commands to do the required functions.



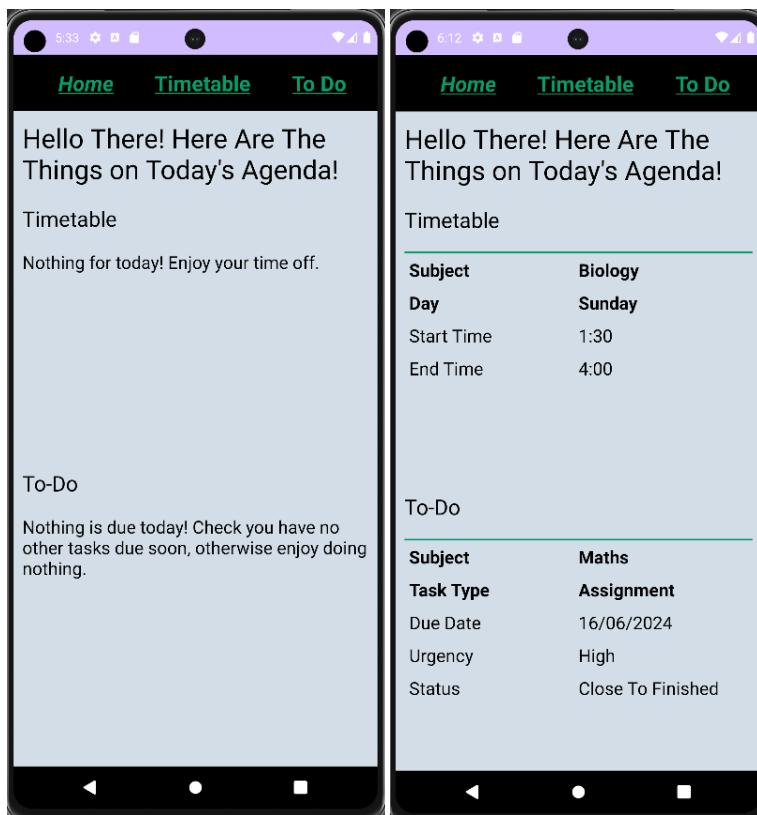
Implementation

Below are screenshots of the final application, changes that I made to the design during development, and the challenges that I faced. I will also be discussing the process of adding another language.

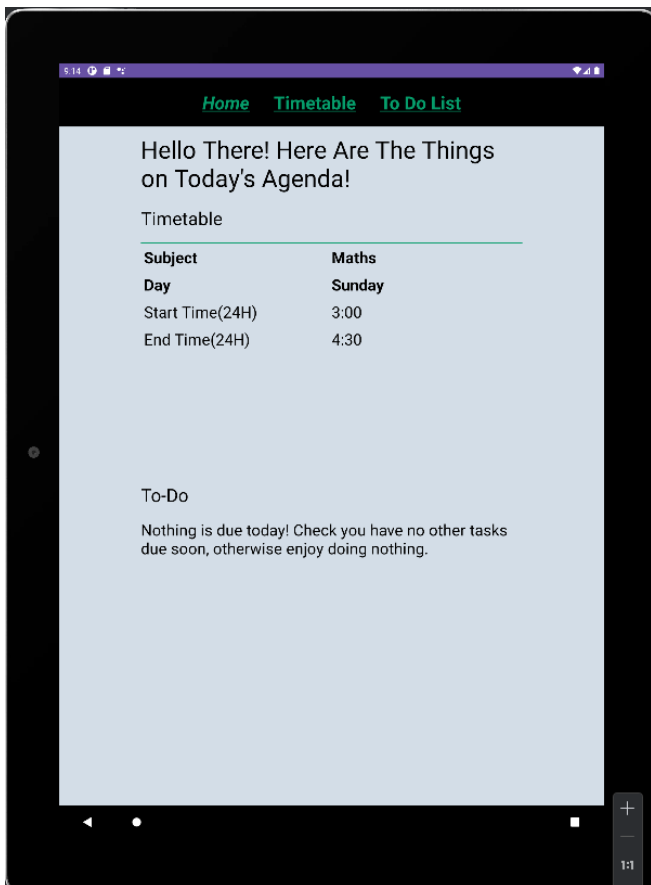
Activity – Home

- Replaced calendar-grid with RecyclerViews for timetable periods due to time constraints and implementation difficulty.
- Timetable periods for the day are displayed in order of start time.
- The two areas displaying RecyclerViews have equal weight, sharing layout space equally.
- Both areas are scrollable, ensuring they do not affect each other regardless of the number of items.
- If there are no timetable periods or tasks for the day, messages are displayed in their place. Navigation indicates the current activity by displaying the text link in italics.

Phone

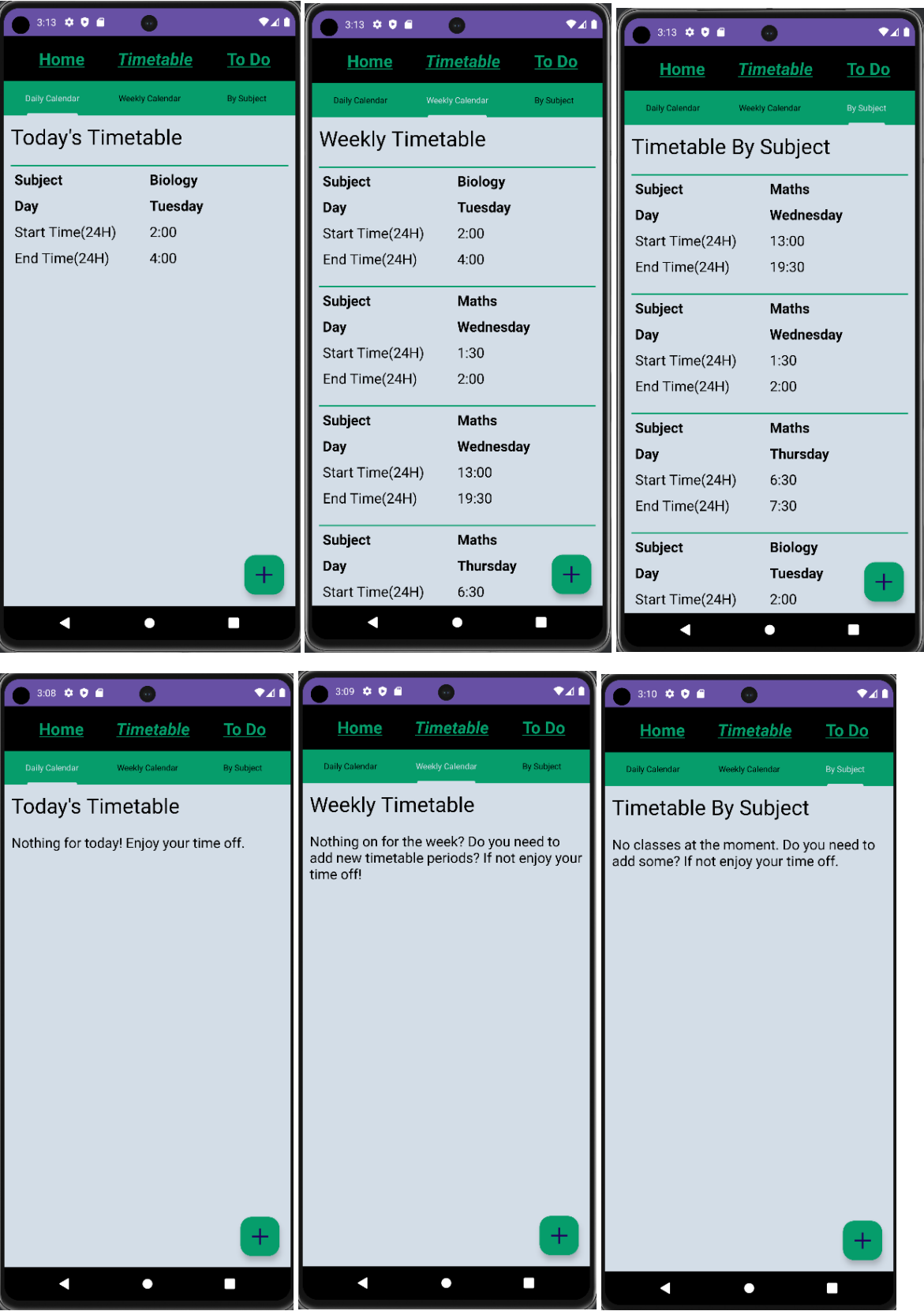


Tablet

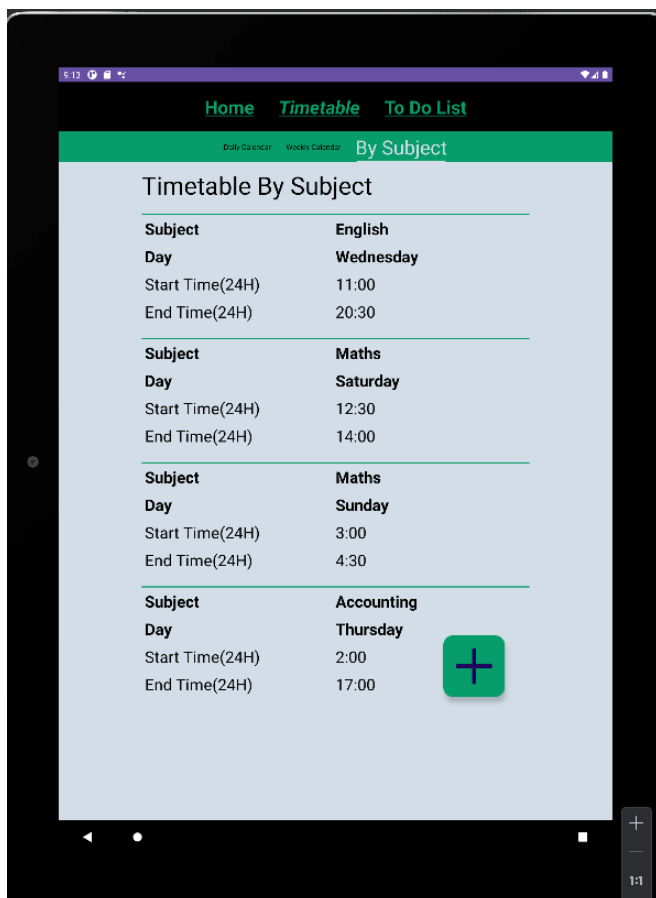
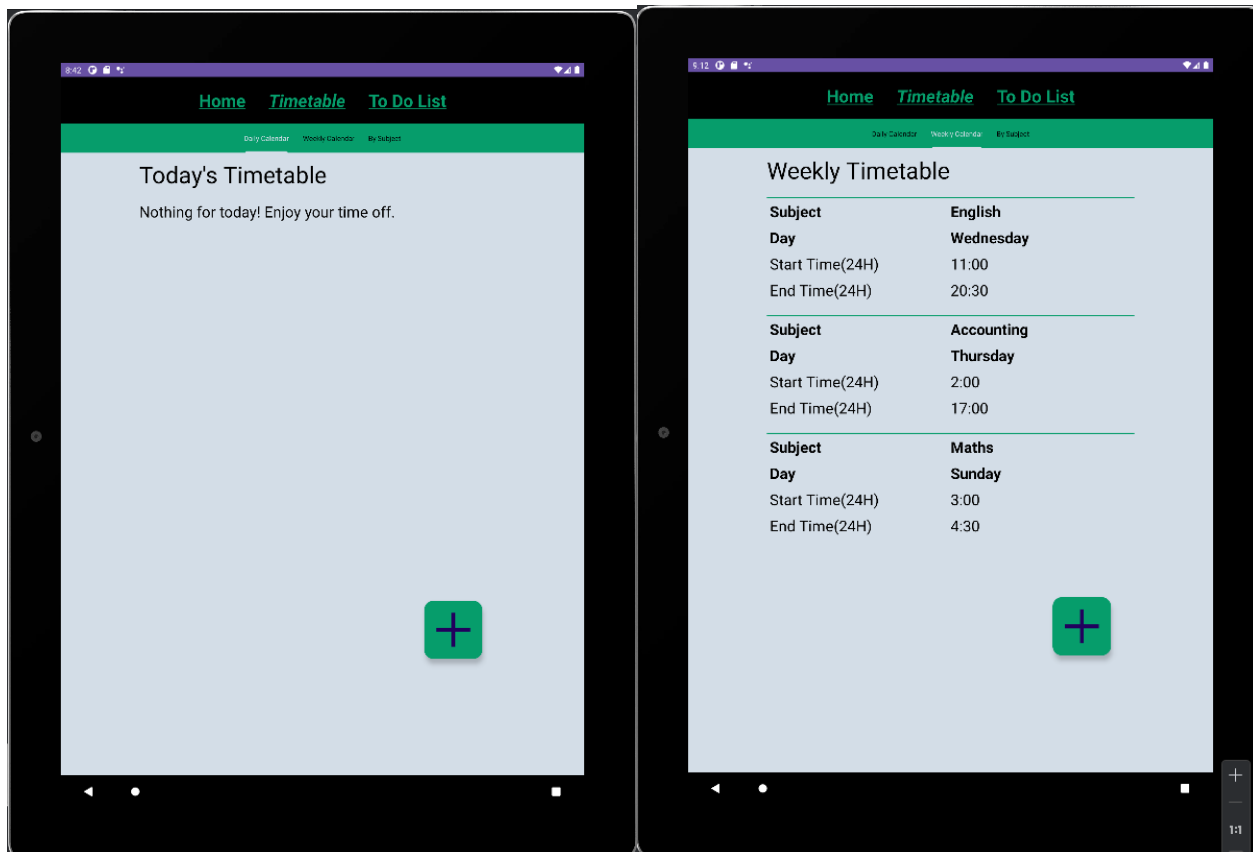


Activity – Timetable

- Replaced calendar grid with RecyclerViews in the daily timetable fragment and weekly timetable fragment due to configuration issues.
- Sorting:
 - Daily timetable fragment: by start time, then end time.
 - Weekly timetable fragment: by day, then start time, then end time.
 - By subject fragment: by subject, then by day, then start time, then end time.
- Strange issues with tab layout text:
 - Unable to configure the text size.
 - Clicking on the by subject tab enlarges it, but this does not happen for other tabs.
- Issues with FAB:
 - The plus icon is set as black in attributes but shows as purple.
 - Unable to make the FAB circular, even though I was using a circle icon.
- If no timetable periods are available, messages are displayed instead.



Tablet

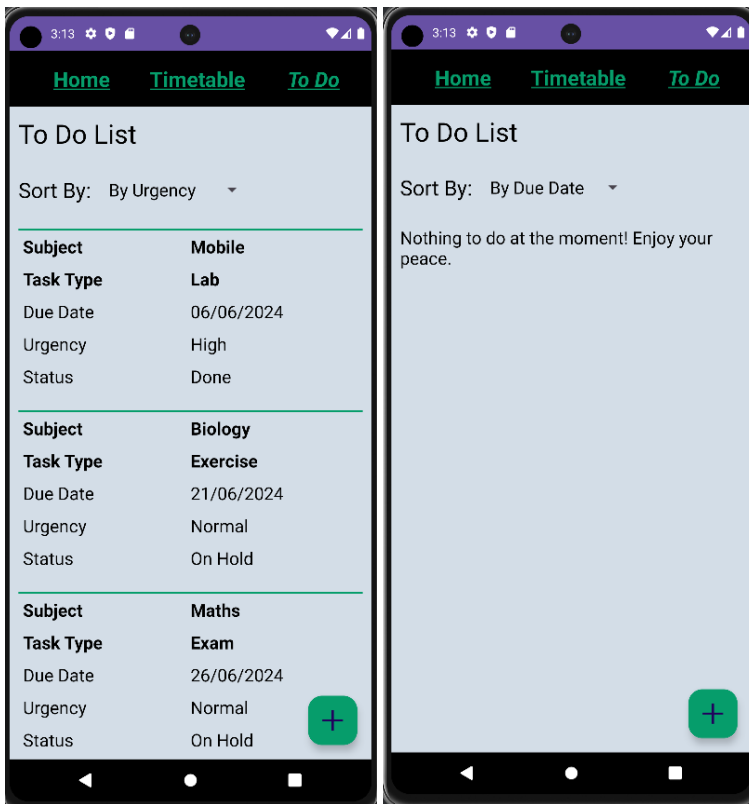


Activity – To-Do List

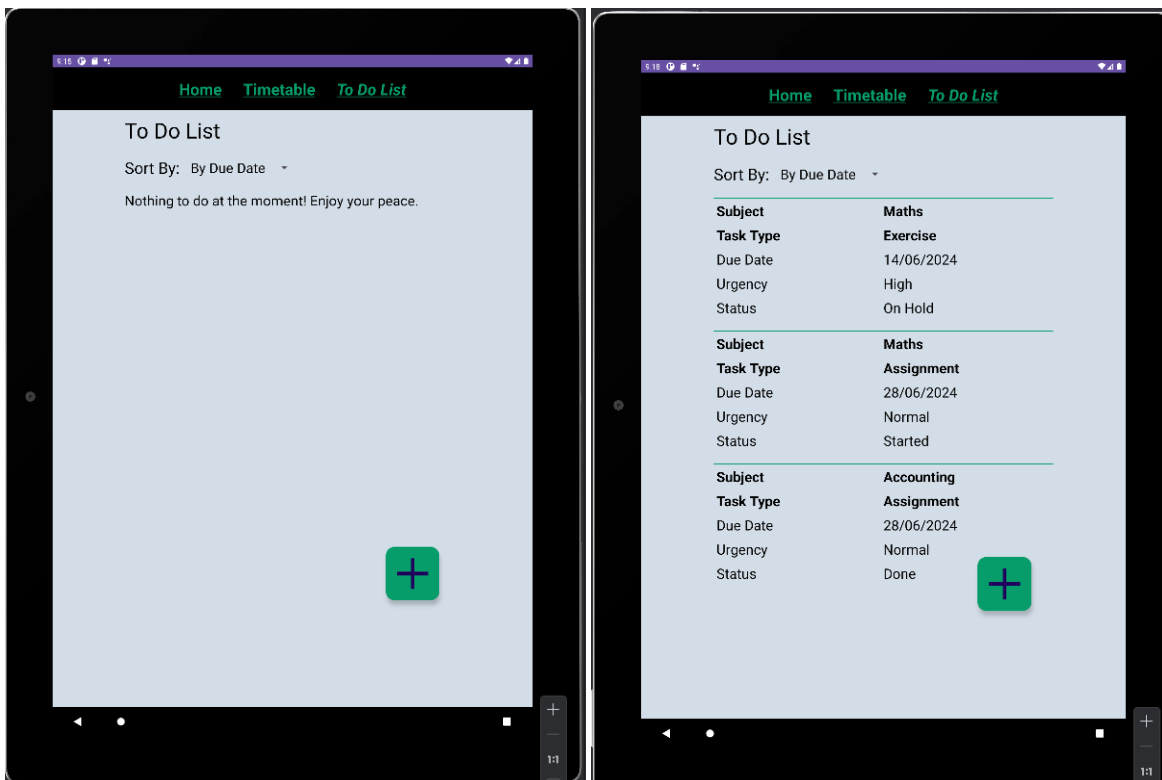
- The to-do list activity matches the wireframe.

- Issues with the FAB are the same as in the timetable activity.
- The spinner provides options to sort tasks, with the default being by due date.
- Hierarchy: due date, urgency, status, subject, and task type. Each sorting option has a slightly different method.
- If no tasks are available, a message is displayed instead.

Phone



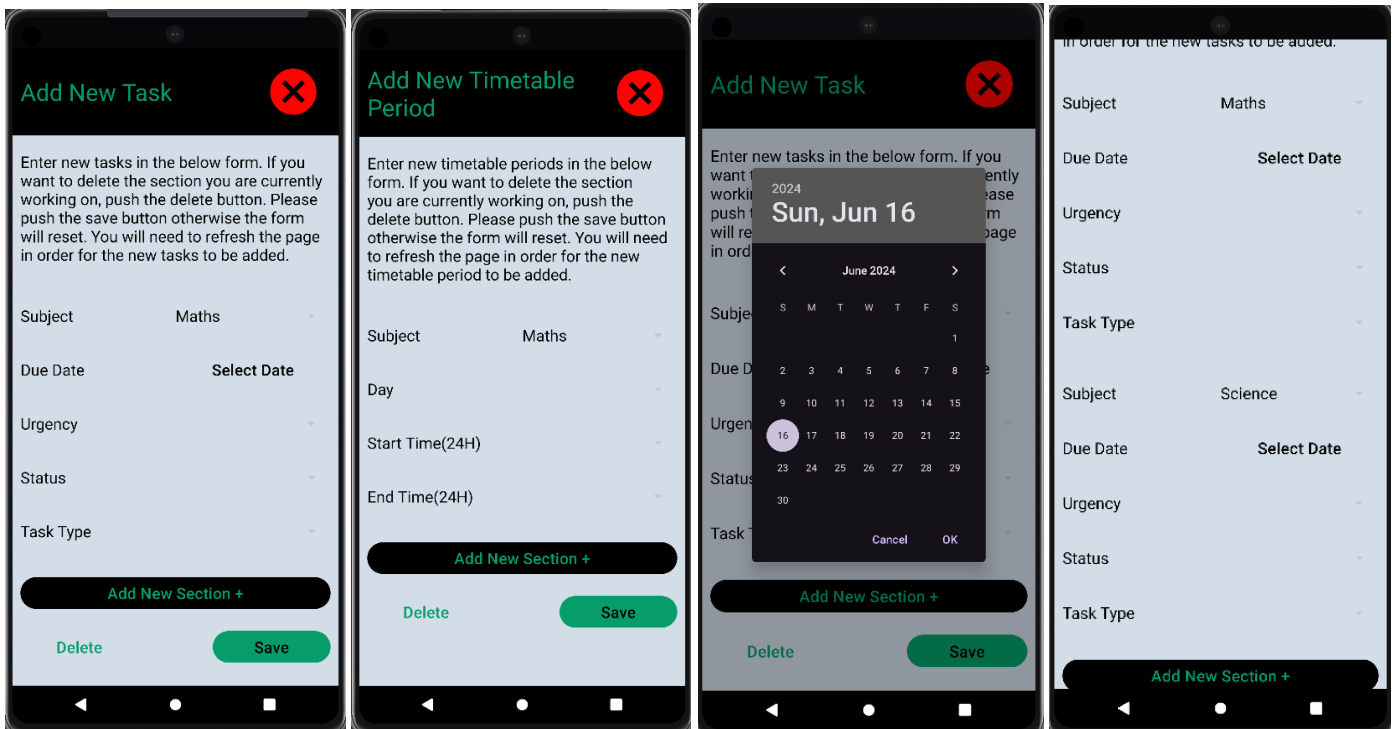
Tablet



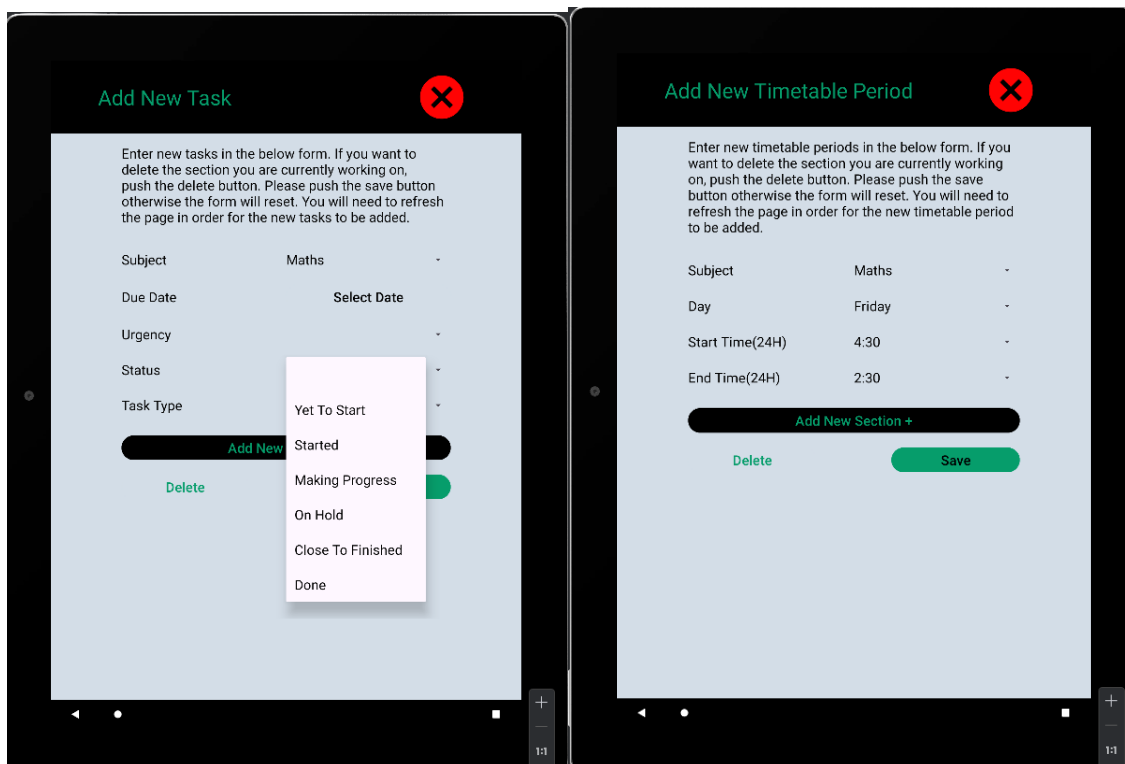
Fragment – Add

- Two fragments for entering data:
 - Add timetable fragment
 - Add task fragment
- Fragments pop up when FABs in their respective activities are tapped.
- Buttons:
 - Cancel button closes the fragment without saving.
 - Add new section button extends the form for adding multiple items.
 - Delete button deletes the most recent section.
 - Save button adds everything to the database.
- Main issues:
 - Snackbar warning for required fields only works for the most recent section.
 - Previous sections with missing required data do not trigger the snackbar, resulting in incomplete data being saved.
 - Unable to fix this issue due to time and skill constraints.

Phone



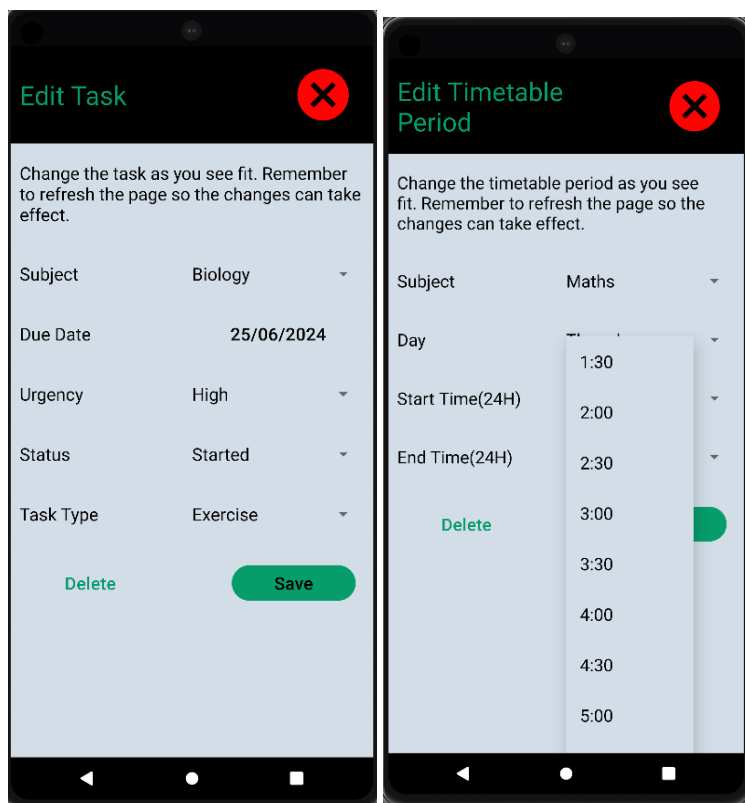
Tablet



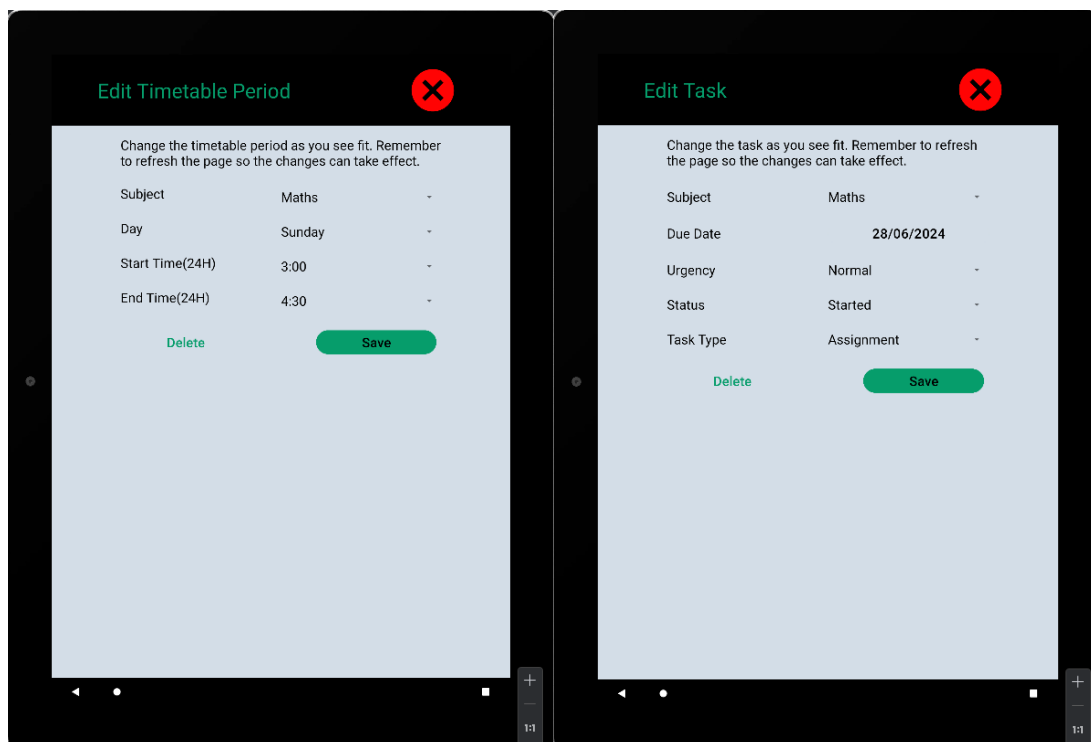
Fragment – Delete/Edit

- Two fragments for deleting and editing data:
 - Edit timetable fragment
 - Edit task fragment
- Fragments pop up when a RecyclerView item is double-tapped in their respective activities.
- Forms are pre-filled with the item's data for easy editing.
- Buttons:
 - Cancel button closes the fragment without making changes.
 - Delete button removes the item from the database.
 - Save button updates the item in the database.
- Snackbars appear if required fields are not filled in.

Phone



Tablet

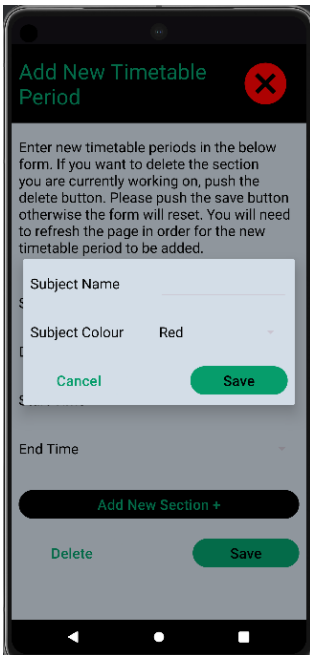


Fragment – Add Subject

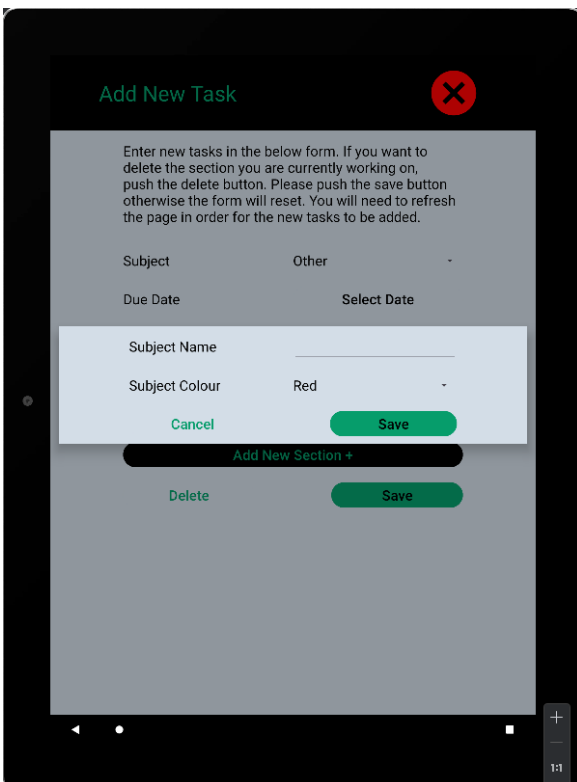
- When a user first opens an add fragment (task or timetable period), an add subject fragment will pop up.
- This is because a subject is needed to create a new timetable period or task.
- Initially, "Other" is the only subject in the list.
- If "Other" is selected, the add subject fragment automatically appears.

- Add subject fragment:
 - Allows the user to type in a subject name.
 - User can pick a color for the subject (intended for calendar-grid use).
- Buttons:
 - Cancel button closes the fragment.
 - Save button adds the subject to the database.
- Differences between phone and tablet:
 - Greater horizontal padding in the tablet layout.
 - Different font sizes.
- No functionality to edit or delete subjects due to an oversight.

Phone



Tablet



String Resources

To implement string resources, whenever I needed to have a piece of text for a label or message, I would just type it in plain text without actually adding it to the strings.xml file. It was not until I finished developing the app that I went through all of the XML files and turned all of the text into string resources with names that were brief but descriptive.

The process of adding a translation was relatively simple. I used the built-in Translations Editor to get the list of string resources and added Spanish as a language, and then one by one I would copy the English text into Google Translate, copy the Spanish translation, and paste it into the Spanish column. Once I was done I got Santiago Cavalitto (whose native language is Spanish) to read over my translations to ensure that there no major errors, and changed the emulator's system language to Spanish to check that it worked.

RecyclerViews

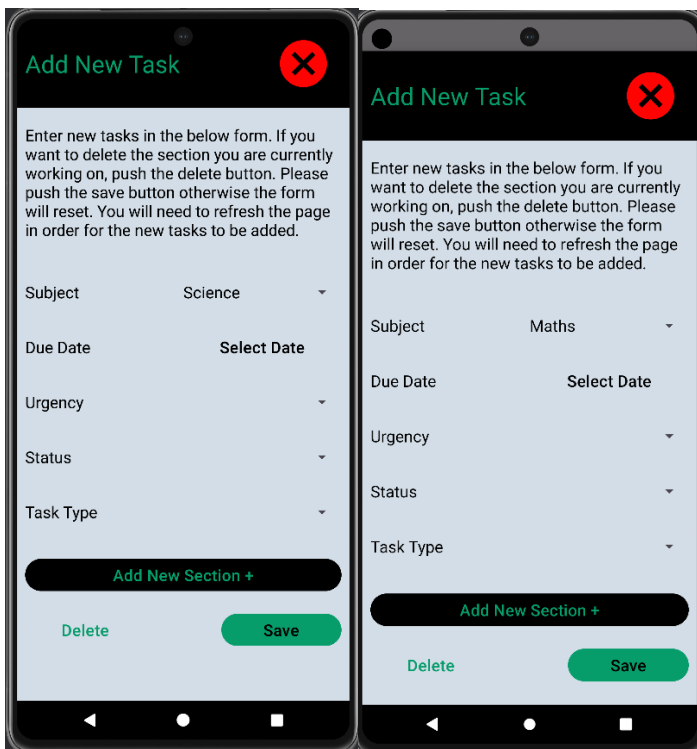
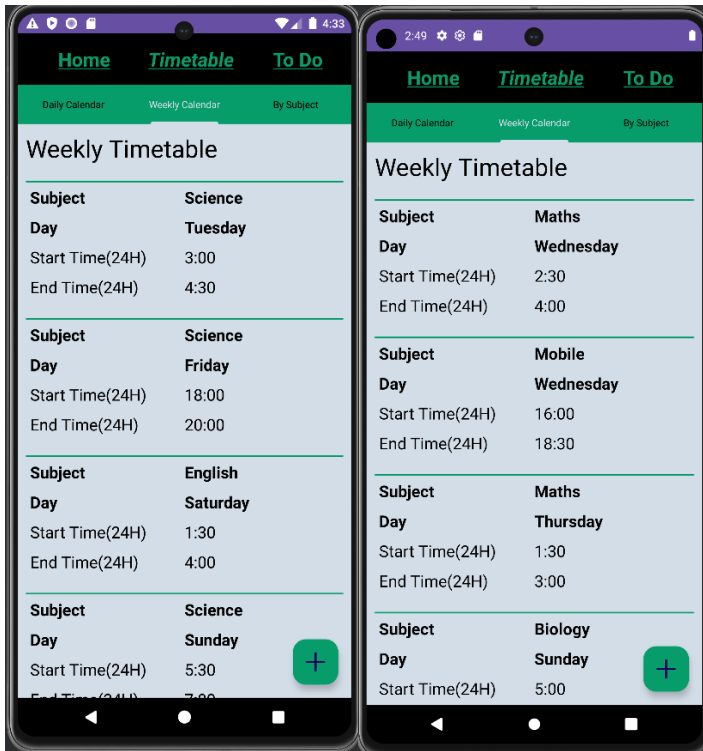
To implement RecyclerViews in my app, I first created XML layout files called task_item and timetable_item and so I could position how I wanted the data to be displayed. The Java classes TaskAdapter and TimetableAdapter extend RecyclerView.Adapter which acts as a bridge between the RecyclerView and the database. The adapters create ViewHolders which are what manage the views (e.g. TextViews) within each item and then sets the data for them. There are three main methods relating to ViewHolders:

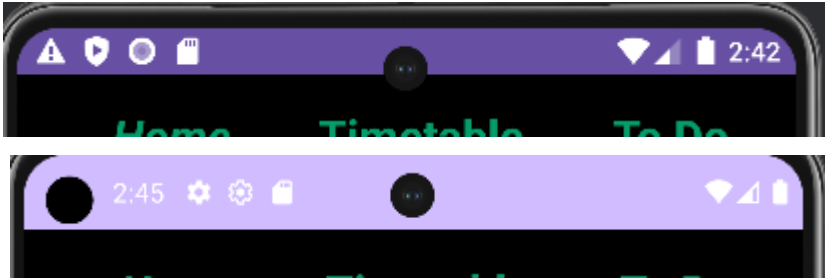
- onCreateViewHolder() which the RecyclerView calls whenever a new ViewHolder needs to be created. It creates and initialises the views.
- onBindViewHolder() which the RecyclerView calls to fill in the ViewHolder with data.
- getItemCount() which returns the total number of items in the data set.

Testing

The two versions of Android I have tested my app on are Android 11.0(API 30) and Android 8.0(API 26). The devices I have tested these on are the Pixel Pro 7 phone and the Pixel C tablet.

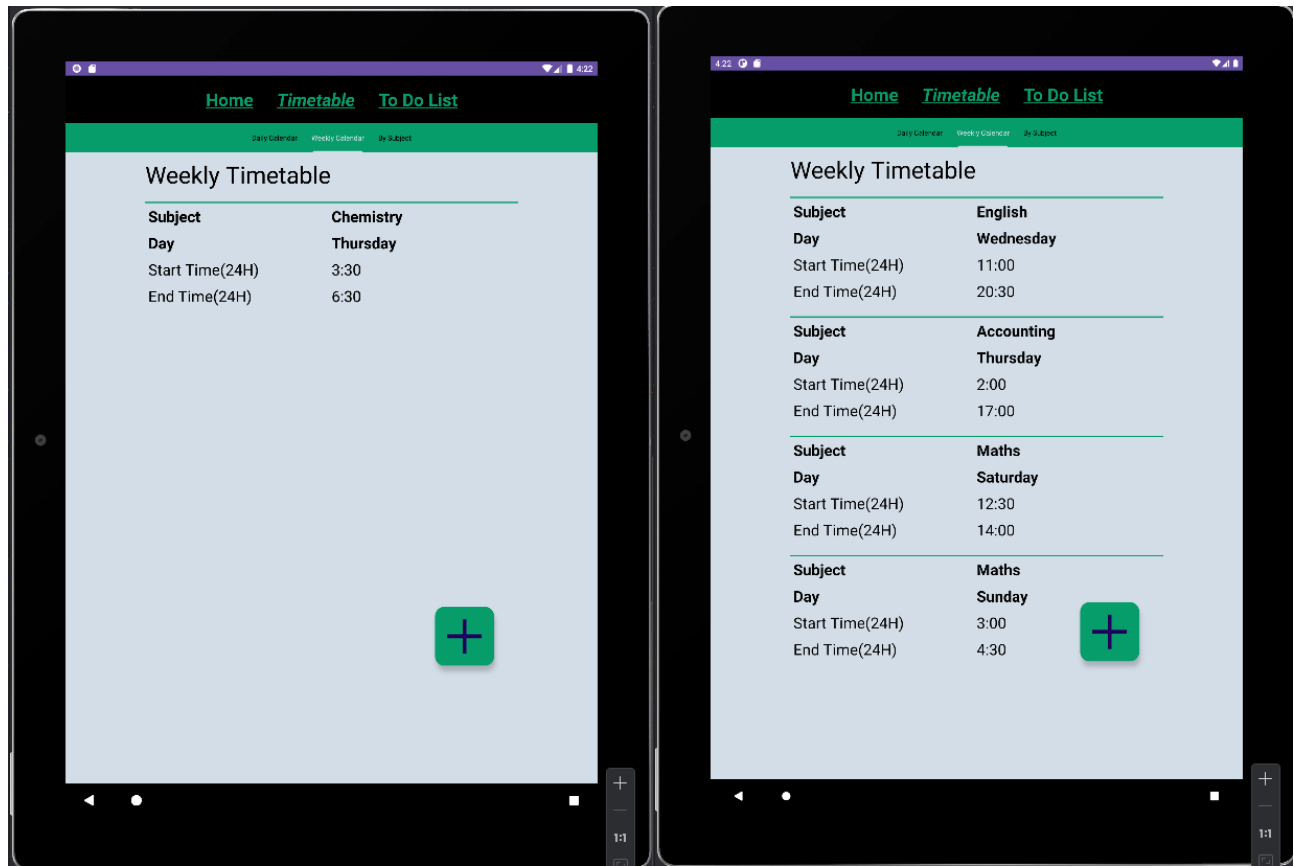
Pixel Pro 7 8.0 vs Pixel 7 11.0

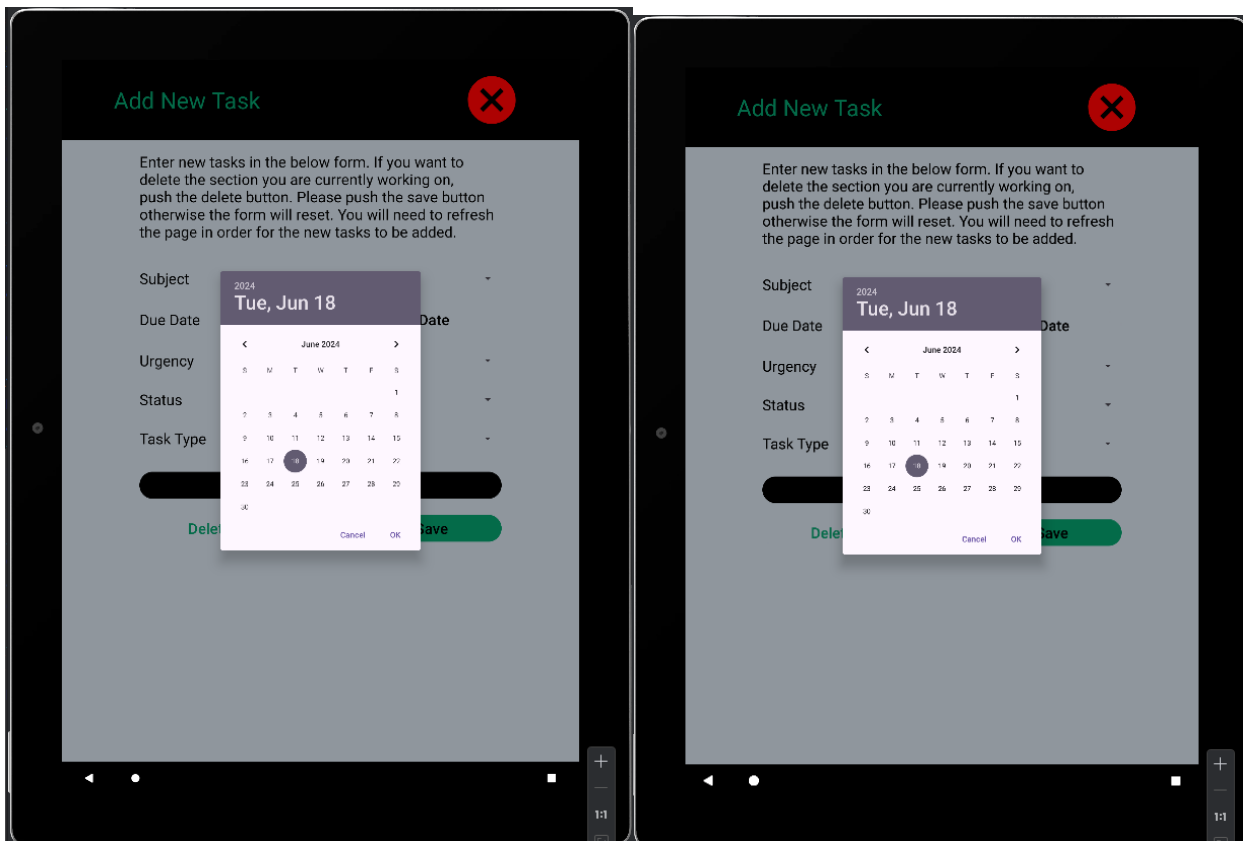




No major differences, other than the 8.0 device was very slow.
 Although they are the same device there is a front camera in 11.0 which as seen above increases the height of the notification strip and therefore slightly decreases the display size.

Pixel C 8.0 vs Pixel C 11.0





I could find no differences between the tablets, the functionality was the same and all the views displayed the same. They even had the same issue regarding the increase of the By Subject tab text size.

Phones vs Tablets

As stated above, there are barely any differences between the 8.0 and 11.0 phones, and between the 8.0 and 11.0 tablets. The difference between the 8.0 phone and the 8.0 tablet is the same as the difference between the 11.0 phone and the 11.0 tablet.

This shows that the design and functionality of my app are robust enough to be consistent with other device sizes and Android versions.

Conclusion

This app fulfils my goal of providing a central area where users can monitor and update all their important tasks and classes efficiently. It is accessible on different Android versions and adapts well to various devices, ensuring a seamless user experience. The app adheres to Material Design 3 guidelines and follows best practices, ensuring users have an enjoyable and positive experience.

The functionalities and features of the activities and fragments, centred around the database and CRUD operations, make data management efficient and straightforward. The use of RecyclerViews provides a flexible method for displaying large amounts of data effectively.

Although I am content with the app's overall outcome, functionality, and adherence to the initial wireframes, I am disappointed that I could not implement the calendar-like feature for the timetables. I hope to develop the necessary skills to add this feature in the future.

References

- Coursera Staff. (2023, November). *What is time management? 6 strategies to better manage your time*. Coursera. Retrieved 5 May 2024 from <https://www.coursera.org/articles/time-management>
- Google. (n.d.). *Material Design 3*. Material Design. Retrieved 6 May 2024 from <https://m3.material.io/>
- tutorialspoint. (n.d.). *SQLite-AUTOINCREMENT*. Retrieved 9 May 2024 from https://www.tutorialspoint.com/sqlite/sqlite_using_autoincrement.htm#:~:text=SQLite%20AUTOINCREMENT%20is%20a%20keyword,used%20with%20INTEGER%20field%20only
- Android Developers. (2024, June). *Create dynamic lists with RecyclerView*. Retrieved 10 June 2024 from <https://developer.android.com/guide/topics/ui/layout/recyclerview>
- Lars Vogel. (2017, September). *Using the Room framework as SQL object mapping library*. vogella. Retrieved 9 June 2024 from <https://www.vogella.com/tutorials/AndroidSQLite/article.html>
- Admin MindOrks. (2019, January). *Android SQLite Database in Kotlin*. MindOrks. Retrieved 10 June 2024 from <https://blog.mindorks.com/android-sqlite-database-in-kotlin>
- CPZackPacker. (2023, May). *CodePath Android Cliffnotes*. GitHub. Retrieved 10 June 2024 from <https://guides.codepath.com/android/Passing-Data-Between-Activities#sharing-data-with-intents>
- Stack Overflow. (2021, July.). *How to create a Custom Dialog box in android?*. Retrieved 28 May 2024 from <https://stackoverflow.com/questions/13341560/how-to-create-a-custom-dialog-box-in-android>
- Android Developers. (2023, March). *Fragments*. Retrieved 28 May 2024 from <https://developer.android.com/guide/components/fragments>
- Android Developers. (2024, June). *Dialogs*. Retrieved 9 June 2024 from <https://developer.android.com/guide/topics/ui/dialogs>