LectureMate

Final Draft Project Report

Manpreet Singh Bance — 33552762

May 2020

Contents

List of Figures

List of Tables

Acknowledgements

I would like to express my gratitude towards my supervisor, Golnaz Badkobeh, for her continued support and guidance throughout the course of the entire project, providing assistance from preparing the project for development to project completion.

I would also like to thank my family for their support in my studies throughout the duration of this degree.

Finally, I would like to thank Lahcen Ouarbya, and the Department of Computing at Goldsmiths, University of London, for the opportunity to work on this project.

Chapter 1

Introduction

LectureMate is designed to be a cross-platform mobile application for use in an educational setting, namely, lectures - whereby, students and other academics can use the application to make dynamic, powerful and relevant notes which are linked to a specific timestamp or set of timestamps. The type of notes which can be made are rich text, uploaded images or camera photos, links, documents/research papers - all of which would the user would deem to be relevant to the point of the audio recording which is being made. As well as this, the user has the ability to highlight certain segments in the audio which they feel are important and they would benefit from later on when coming back to review these notes, for example, during revision.

As previously mentioned, the mobile app is mainly aimed at educators and academics in the educational setting but due to its capabilities could potentially be used in a range of other settings. Primary research showed that there were no existing applications on the market which offered the functionality that this mobile application aims to do. Existing applications such as Evernote are based primarily on text-based notes with the ability to add further attachments, whereas LectureMate would be audio-based allowing for notes to be made based on the timestamp in the audio recording.

Secondary research was able to prove that educators (the primary target market) and among students at Goldsmiths College, University of London, that there was no one consistent form of making clear, relevant notes and that the target market relied on a number of various applications to make notes which was not a practical solution. These applications incliuded the likes of Evernote, Microsoft OneNote, Google Docs, as well as traditional notes which were made with pen and paper which were later difficult to merge with existing notes on these applications.

From this, it was clear that the application would aim to provide a satisfactory solution to the needs of the market and honoring what is in need by the target market which was able to be identified through the research. As a result, upon deployment of the application we would be able to see whether this is fulfilled and is versatile enough to meet the needs of its users. This is evidence to show how an application and platform such as LectureMate is vital in filling a much needed gap in the market.

1.1 Motivation

The motivation behind this mobile applcation is upon research the lack of platforms which allow users to make notes based on audio recordings. As well as this, in first-hand experience, I was able to identify that I found note-taking particularly inconsistent and upon coming back to these notes to revise I found these notes to have no context and difficult to understand. Therefore, I carried out some research to see whether any platforms existed which allowed notes to be made but with some context and found that there were none for wide use.

By this and by communicating with peers among different educational settings, I was able to recognise that most univerities and lecture theatres had recording equipment. I initially upon ideation, thought of pursuing this, however, later came to the conclusion that this would not be a viable option due to the issues with obtaining the permission of those individuals who may be visible in the recording, and as a result I made the final decision to proceed with the same idea however, using audio recording instead.

1.2 Scope

LectureMate aims to combat the issues that students and academics face in the note-taking and revision aspect of their learning by providing a platform where they can make clear, dynamic and consistent notes in order to tackle the existing problems faced with making notes where the notes are made across different platforms making them different to compile and are inconsistent. Therefore, making clear, consistent and dynamic notes using LectureMate are a solution to these issues.

1.3 Objectives

One of the first things which were looked at was how we can meet the objective to fulfill the need for the target market which was identified. LectureMate does this by providing a platform to satisfy the need for making clear notes.

We needed to make sure that the application met the needs that our target market fed back to us when conducting our primary and secondary ressearch. Another objective that was indicated through executing this research was that the application would have the ability to highlight specific points within the audio where upon reviewing, would be able to identify the key, relevant sections of a lecture or recording.

The security and integrity of the notes which are made are a priority as these notes are to be kept safe whereby unauthorised changes should not be made to the notes as well as their security being ensured as to them not becoming deleted or corrupted. Therefore, considering this, the app aims to be a safe and reliable platform for its users so that they do not need to worry about the integrity of their notes. The application will do this by saving the note as a file in a folder creating regular backups each day or manually.

As the files will be stored locally on the device, the user is responsible for any data which might be subject to any tampering or corruption to the files when being manipulated through any third party application such as a file explorer. Using the application to make any changes to any notes will be a tested and safe way for users to alter notes.

Chapter 2

Technical Specification

2.1 Background Research

Here, I will focus and discuss the research I carried out prior to commencement of the design of the application. I will begin technical research looking at the potential for technologies I may use to develop the mobile application and briefly address the reasons behind this.

2.1.1 Existing Systems

In the research stage, I wanted to explore the market for existing applications which perform or complete similar or exactly the functions that I was aiming for in this application. I decided to look at features I would be able to implement from others or simply known, working models which wre proven to be good applications and have relevant, useful features I could also implement. However, I wanted to be inspired by these and not just implement the feature in the same way as the app I would be borrowing from.

Soundcloud

[?] While doing research, I thought of existing systems which I can obtain certain features from which I might be able to incorporate into the mobile application. Soundcloud came to mind whereby in their interface, they have the ability to have their users to add comments at certain time points in the audio which had been uploaded. I thought this same model would be able to be applied to this application as notes can be added at a specific point in the audio recording.

Evernote

Again, whilst looking at exisitng applications, if not one of the biggest, is Evernote who provide a cross-platform note taking solution. I therefore studied this application well and listed features I also look to implement into this application - these included: the ability to upload images, audio attachments and links. These features would all be very useful in a note taking setting, especially in a lecture setting.

2.2 Overview

The technical architecture of the mobile application is as simple as the layering mode. The logic of a layered architecture allows the separation of the Front-End, Middleware and Back-End, however still allowing for cohesion between these, which will be detailed further.

The uppermost layer will be the presentation layer - which will be the appearance and interaction side of the application, the middle layer - which will be responsible for the (business) logic behind the application containing the code behind the actions interacted with in the presentation layer, and the data access layer - which will be the back end code which enables the user to access and make changes to the notes, although stored locally, they will still carry out CRUD functionality similar to that used in databases.

2.3 Application Frameworks

A framework is an abstraction where software is written to provide generic function to achieve a task or tasks. This mobile application has made use of some frameworks to contribute to the functioning of it - mainly being in the Front-End or 'Presentation Layer'.

2.4 Full Stack Development

As previously mentioned, the web application was designed and structured in a way conforming to the stack structure: Front-End, Middleware, and Back-End. This section will further discuss the specification of technologies in each tier of the stack and the reasons behind their uses:

2.4.1 Flutter

The entirety of the mobile application will be using the existing Flutter framework which is an open-source UI software development kit created by Google - used to develop applications across mobile, web and desktop applications. I will detail the framework architecture and the way each of these are used in the development of the mobile application:

Dart platform

Flutter apps are written in the Dart language and make use of many of the language's more advanced features. The Dart language package allows during development for implementing features into the application which help shape the appearance and layout as it appears to be.

During the development stage, the Dart package also allows for dynamic and a live coding experience which involves hot reloading whereby any changes in code can be updated and made live without having the whole application have to be restarted for the new changes to take effect which was a welcome and proven helpful feature to the process.

Flutter engine

The flutter engine provides low-level rendering support interfacing with iOS and Android platforms which hosts the Fluttter applications and manages the way they look and behave controlling and making use of its core libraries which include animation and graphics, file and network I/O, accessibility, plugin and runtime and compilation toolchain. However, these were things which during the development process were something which were not deeply explored and only utilised for basic functionality.

Using the Flutter framework allowed for a more reactive, modern framework for creating a modern and engaging mobile application which would also function well.

Foundation library

The foundation library written in Dart allowed for the use and implementation of basic classses and functions in the mobile application enabling access to APIs to communcate with the engine and construct the application.

Widgets

Flutter widgets are built into the package and work as the different components which make up together to build the application, such as buttons and text. These are included as standard and are used in the application for some of the components.

Design-specific widgets such as Material design using Google's design language as well as Cupertino which use Apple's Human Interface Guidelines iOS design, both make use of these respective design langauges to display the application best in their respective, native environments to give them a more uniform look despite being on different platforms.

Google Firebase

At a later stage in the implementation stage, it was concluded that issues would arise with the storage and retrieval of data, namely media such as images, videos and audio as it would be difficult to reference in using the application and it made sense to make use of a database to solve this. Due to the nature and cohesiveness due to being both Google developed systems, it would make sense to use Google Firebase as the database management.

This would allow for the storage of raw data such as strings and numerical characters; Firebase also has a solution for the storage of media items as this utilises Firebase storage which gives each record a unique identifier which can be associated to the respective note and would could be stored successfully.

Chapter 3

Project Management

3.1 Labour/Resource Management

Prior to commencement of the project, I planned to allocate resources in the form of time, being the number of hours spent, on each stage of the project with ideation to conception, research, development and testing stages to ensure that this was all done in good timing and enough resources were spent on each stage so that it would be produced to a high standard. Following the allocation of resoruces available throughout the project, I created a timeline with set provisional dates whereby these tasks should be completed, referred to as milestones.

Taking this approach to the allocation of labour and resources means I can plan when certain tasks should be done and if these are not met, I can allocate further resources to make the project less susceptible to interruptions, which may include:

1. **Overlap of tasks:**
   The tasks were divided in smaller segments to ensure better productivity and efficiency in completing each tasks and to make sure that each task was completed correctly prior to moving onto the next task. This guaranteed that there was less of a feeling of a being stuck on a particular task for a prolonged amount of time and a wider variety to provide a faster workrate of tasks being completed. Due to the nature of the developmemt process being in Flutter, the bulk of the project was the middleware and backend controlling the way functions and features were supposed to behave and whether these were being suitably met. The tasks were divided in such a way that there was no dependency of completion of a task to complete the next as such until it came to testing to see whether these worked together cohesively.

2. **Time keeping and meeting deadlines:**
   The time keeping and milestone aspect of the planning process was very important so that each of these were being met on time and no stage of the development process was falling behind so much so that it would not be complete and the application as a whole would suffer. To combat this, I created a blog so that I was keeping track of the developments made in the project each week to track whether this progress was good and steady and would meet the milestones at the correct times. Planning these milestones enables for a greater control of the project and the way it is progressing.

3.2 Methodology

3.2.1 Agile Development Method

The Agile Project Management Methodology was utilised where, by definition, an iterative process made use of at each stage of development in all areas of the development stack requiring constant effective communication. Using the Agile method meant that the project was dealt with the most efficient way possible with larger tasks being broken down and divided as to the larger and smaller tasks taking varying time to complete and complete these tasks effectively.

3.2.2 Kanban

The Kanban method is also used in order to keep track of the items which need to be completed and which are still yet to be completed as well as a section to review code and carry out ah-hoc testing as well as component testing to see whether each component is working as expected. This also provides a more broader look at the tasks to come and time to prepare for these, seeing whether any previous tasks are dependant or feed into the following task. The Kanban method is visualised using Trello.

3.3 Version Control

3.3.1 Git

The project makes strong use of Git and its features for version control purposes, with the ability to see and track any changes to files and have a history of the previous changes made and to apply any future fixes giving more of a sense of control and responsibilty for each task to be completed correctly as well as the a timestamp of when this has been completed to identify whether it has been done in good time according to the milestones and personal proposed dates.

Chapter 4

Analysis

4.1 System Requirements

System requirements outlines the criteria needed to complete the application to a standard meeting the MVP and beyond. This is looked at in two separate sections, functional and non-functional requirements. These requirements are set here so as to build and define clear terms to avoid any misunderstanding in any of the stages of implementation which in turn allow me to predict the challenges I might face and be ready to spend more time and allocate resources here whereby I can plan time spent on each stage to clearly visualise a viable and realistic application to produce in the timeframe and to what standard I can produce tbis to.

4.1.1 Functional Requirements

Essentially minimum requirements the application is expected to handle and accommodate:

ID	Description	Vital/Optional Requirement
FR01	Mobile application to run at least on all Android devices	Vital
FR02	Mobile application requires constant, reliable access to internet connection to user's device	Vital
FR03	Mobile application user input data to be stored securely on Google Firebase	Vital
FR04	Mobile application to back up audio recordings to be stored locally on device in the event of unreliable internet connection	Optional
FR05	Mobile application should allow the user to make notes and be able to retrieve and view these	Vital
FR06	Mobile application should enable the user to carry out CRUD functionality through the database	Vital

Table 4.1: Functional Requirements

4.1.2 Non-functional Requirements

ID	Description	Vital/Optional Requirement
NFR01	Mobile application should run on Android or iOS mobile device	Optional
NFR02	Mobile application should be usable easily for any user to achieve the purpose of the application	Vital
NFR03	Mobile application should store the data with integrity without the ability to maliciously alter data	Vital
NFR04	Mobile application should have all features working as expected without any bugs	Optional
NFR05	Mobile application should perform all actions without significant delay within reasonable time period of no more than 10 seconds	Vital

Table 4.2: Non-functional Requirements

4.2 Stakeholders

Stakeholders are those with an interest in the subject - which in this case is the mobile application. As previously mentioned, the aplication is targeted for those in educational settings including students and academics but can also be used for various functions outside this setting.

Taking this into consideration, the application can be used by people of all ages, however, more than likely those over the age of 18 but is not limited to this. With this assumption in mind it made sense to carry out a survey to see whether this was assumption could be proven true or disproved which is of great importance so the specification of the application can be modified accordingly.

4.3 Use Case

A use case diagram is a illustrated display of intercommunication between users or parts of a system which shed light on the requirements of the system and the role each component plays in order to achieve a successful product/outcome.

4.3.1 Use Case Scenarios

4.3.2 Use Case Diagram

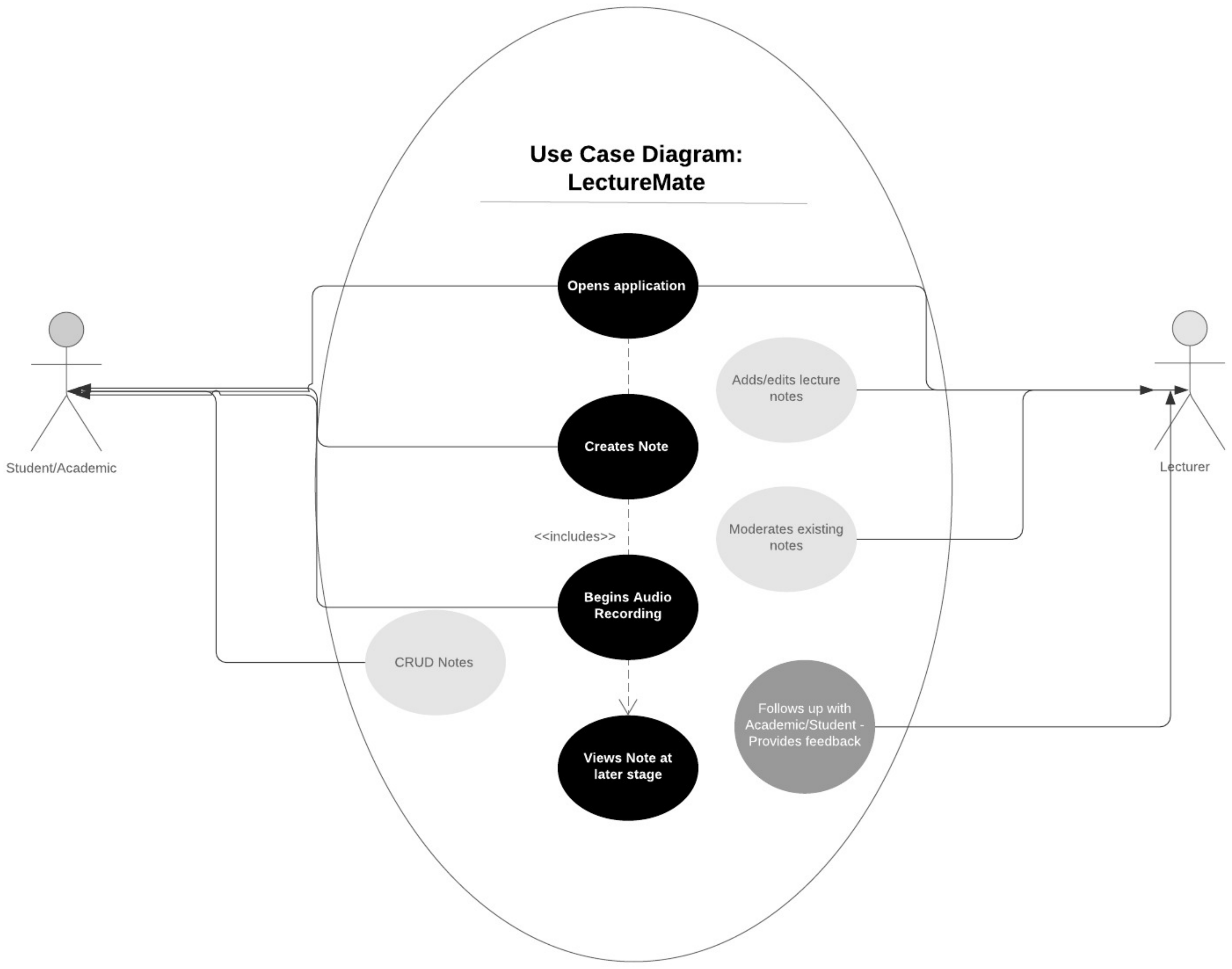


Figure 4.1: Use Case Diagram

Chapter 5

System Design

5.1 Design Models

5.1.1 Activity Diagram

5.1.2 Sequence Diagram

5.2 UI Design

Chapter 6

Implementation

6.1 Front End - Flutter

Front end web development, also known as ”client side programming”, is based on everything that happens in the user’s interface. From the layout, user interface and user experience, it is what the users can see and interact with. Since the project heavily focuses on the user base, it was essential that existing and potential clients feel comfortable with using the application. I established early on in the project that a mobile application would be best as it is the most easily accessible via multiple range of smart phones.

The front end was handled by the Flutter package which makes use of the Material package as part of Flutter. This managed the UI and layout of the application. Due to the nature and simplicity of use of this package, it was very easy to look at making the UI aesthetically pleasing and pretty from the time of implementation - whereas I was expecting to have to focus more on the functionality of the application and then later look at making the UI more user friendly.

6.2 Business Layer - Flutter

The business layer which is the layer which handles the communication between the front-end and back-end operations - this is also written using Flutter as it has the ability to seamlessly link both the Flutter framework and the Google Firebase database.

This consists of the database.dart file which acts as a database helper file containing the functions to submit and retrieve the data to and from the database. As well as this, due to potential unreliable internet connections, offline use has been considered where the audio recording is stored locally in case it has not been possible to upload to the database.

Apart from this the other business layer functions are execute in smaller sections within classes in the code which can be seen in more depth in the Code Review section which proceeds after this.

6.3 Back End - Google Firebase

The back-end is managed entirely by Google Firebase, which as a Google product works well with Flutter which is also developed by Google. It is controlled through script written using the database helper which contains functions to execute CRUD functions in the database.

The database is a NoSQL database which is made up of collections, in the case of this application, a single collection 'notes' which contains documents which is each note uploaded in the application. The format of each note is the Tite, Body of the note, the Audio Recording URL and Recording Duration.

When being retrieved from the database, the application takes the URL of the audio recording and plays this back using the playback feature to add better functionality and ease of use to the users.

6.4 Mobile Application Development

Using Flutter has allowed the development of the application to be written once using the Flutter engine and can be used on both Android and iOS devices with no compatability issues which may depend on the platform.

This is a huge advantage as it allows the application to be deployed to users of both these devices and it can grow rapidly with continuous integration in potential developments in later revisions of the application.

The development and implementation phase was spread over a long period of four to five months whereby I was able to tackle each feature and spend a reasonable amount of time making sure that they would work as expected. Initally, I decided to focus on the functionality over the appearance and UI of the application. However, the ease of use of Flutter allowed the appearance to be worked on easily in a short space of time which working on the functionality.

I would utilise Version Control in the form of Git to manage the code and see the changes I had made to keep track of the progress made as well as have a backup in case of loss of work or to rollback changes which had caused major or significant issues to the application. This was very helpful as it also meant I could review my code and spend time looking at whether any code was redundant or could be made redundant to simplify the code for clarity and speed when compiling the application at each stage.

6.5 Code Review

I reviewed the code I have for the application to explain the workings and how they contribute to the functionality. I will analyse and review each of the most important classes in the code to give a brief but clear understanding of their implementation:

6.5.1 database.dart

As previously mentioned, I have included a database helper file to manage the CRUD operation and functions between the front end application and Firebase database.

The way in which it works is firstly by using the packages to access the Firebase Storage (Firestore) and Database as well as packages used to store the timestamp of the notes (Intl) and to upload files (Dart.io).

Firstly, a new collection reference is crerated where the collection in the database, notes, is referred to and this is initalised so that all functions will access this to execute the operations:

uploadingNote

A condition is set to check whether a voice_note has been added to the note, if not, the title, body and timestamp of the note are uploaded to the database. If an audio recording has been added, this is uploaded to Firebase Storage (Firestore) and the unique id is stored and is called in the upload of the note alongside the other parameters where the audio file URL will also be stored.

getNotes

A List data structure is created to retrieve the list of notes in the database, referred to as a Snapshot. This reference is then used to get all the documents saved in the database, a for statement is then used to itemise each of the notes by the total there are stored and these are returned.

editNote

The notes collection is referenced and the document to be updated is retrieved using the document ID and this obtains the title and body of the note and in the editNote class this is displayed in the Text Editing Controllers to allow the user to make changes to the note.

deleteNote

This function simply deletes the note upon the users decision. Similar to the editNote function, it is referred to by its Document ID and then this is deleted.

6.5.2 newNote.dart

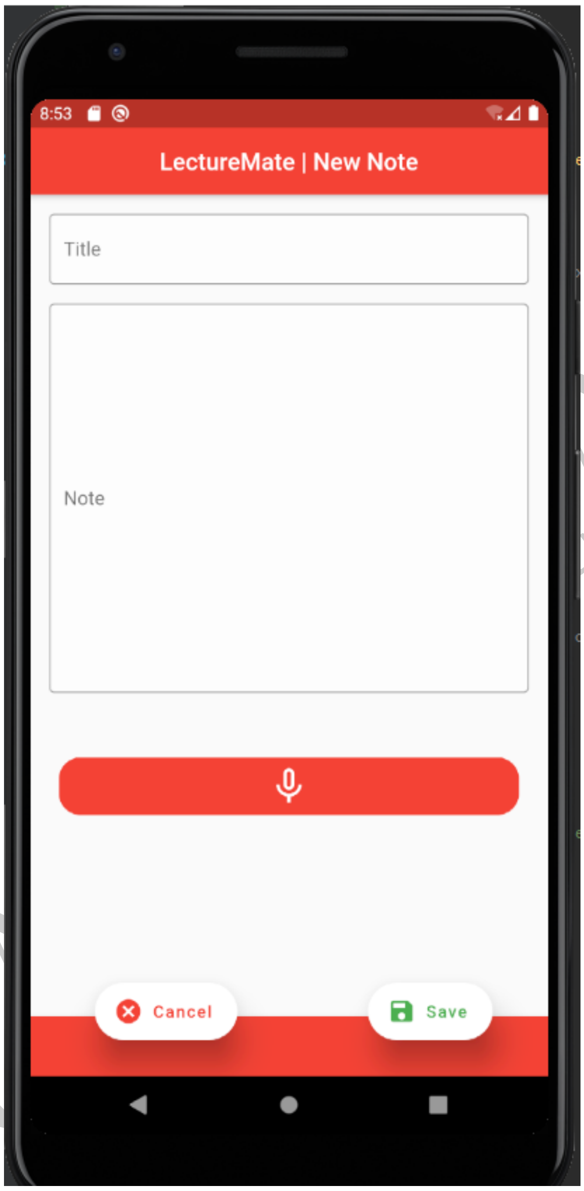


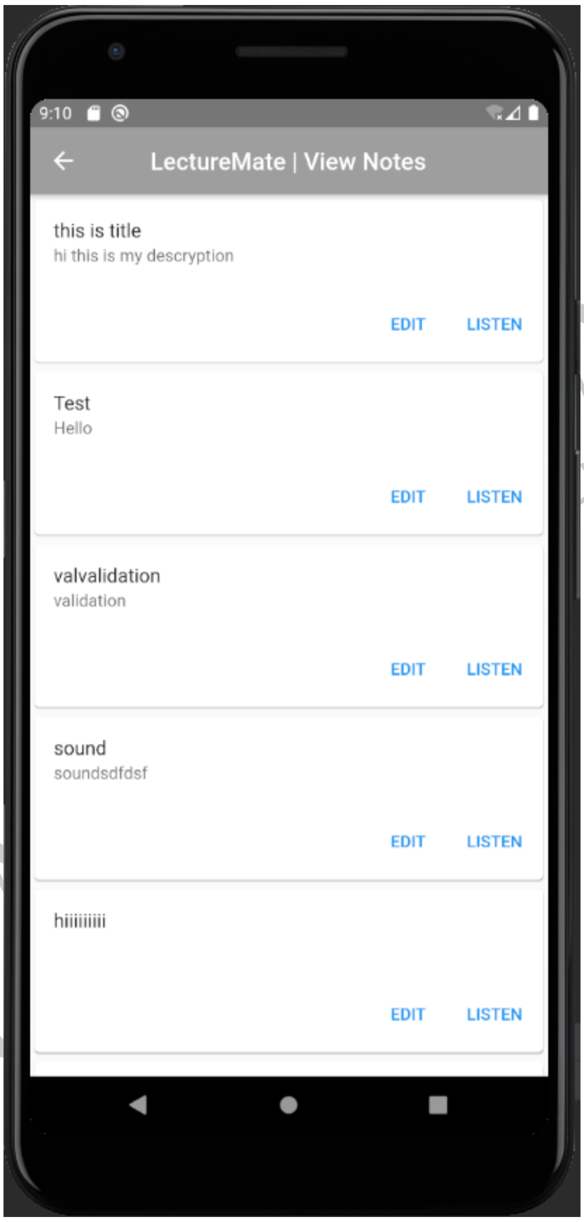
Figure 6.1: New Note UI

This class manages the operations used in creating a new note and the UI on the page.

Features of the New Note UI are the two text editing controllers containing the Title and Body of the note, as well as this, the Audio recording button which on press will begin recording and finally two buttons which cancel the note with a popup dialog for confirmation and a save button to save the data to the database.

This is very similar to the Edit Note UI, however, it does not display the audio recording feature as this only allows for updating and a recording cannot be updated as such, thus only the title and note body text controllers are modifiable.

6.5.3 list.dart


9:10

LectureMate | View Notes

this is title
hi this is my descryption
EDIT LISTEN

Test
Hello
EDIT LISTEN

valvalidation
validation
EDIT LISTEN

sound
soundsdfdsf
EDIT LISTEN

hiiiiiiiii
EDIT LISTEN

Figure 6.2: View Notes UI

6.5.4 playback.dart

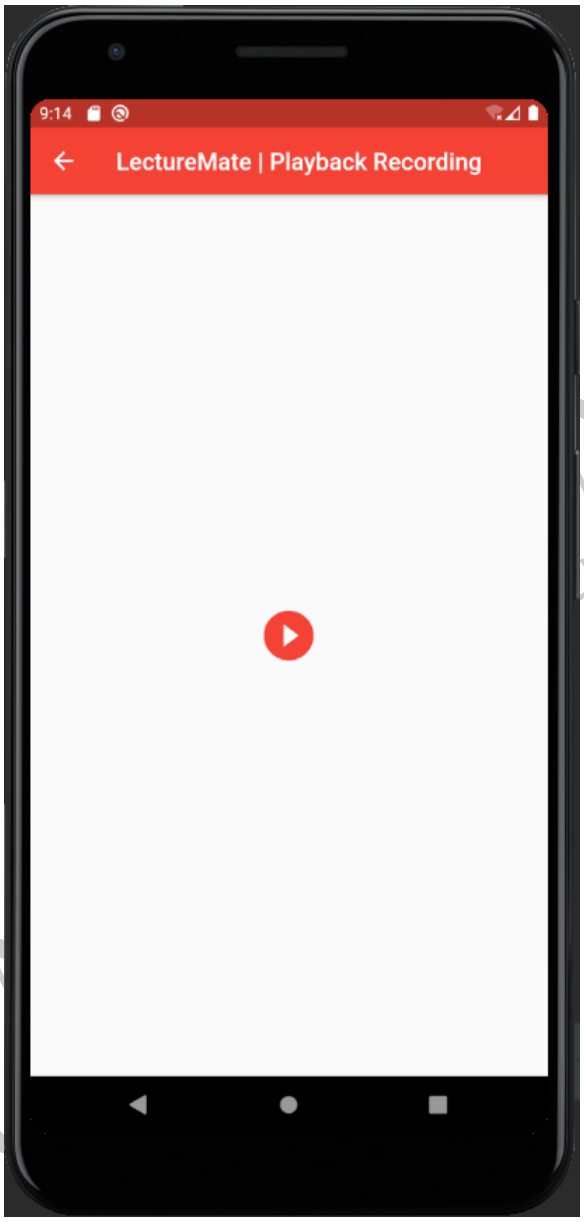


Figure 6.3: Recording Playback UI

6.5.5 voice_message.dart

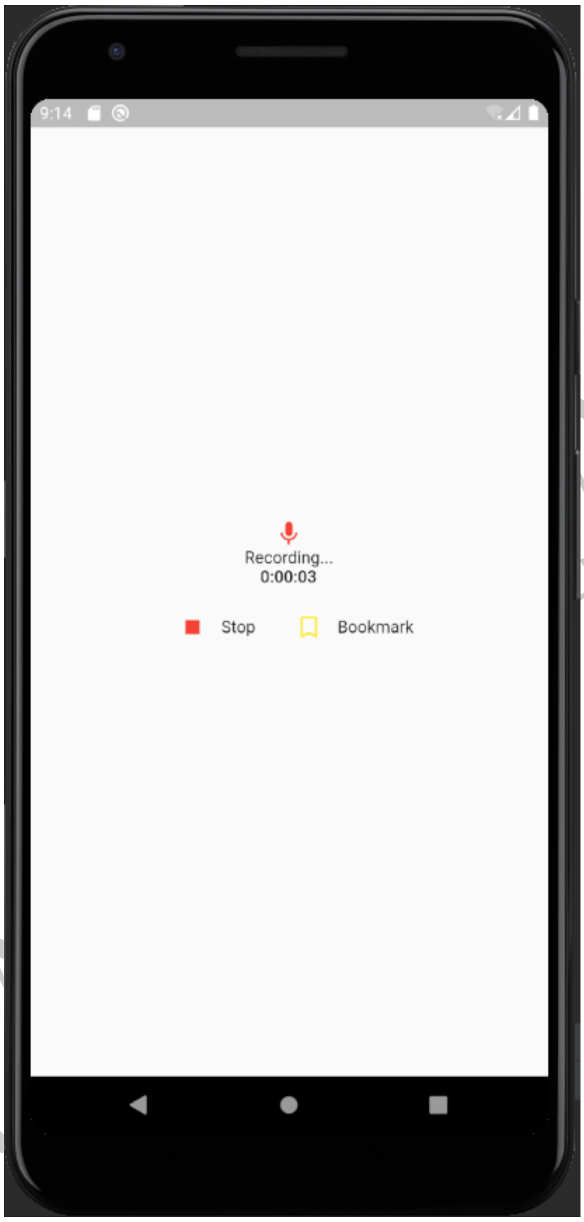
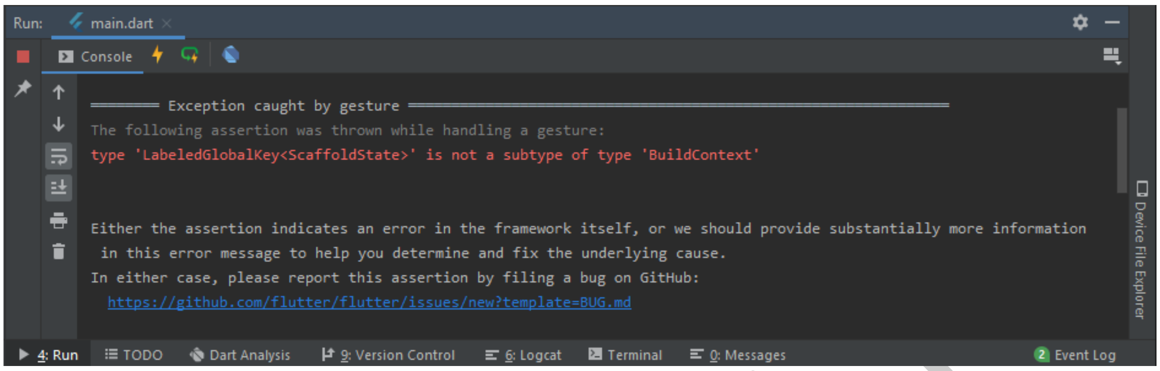


Figure 6.4: Record UI

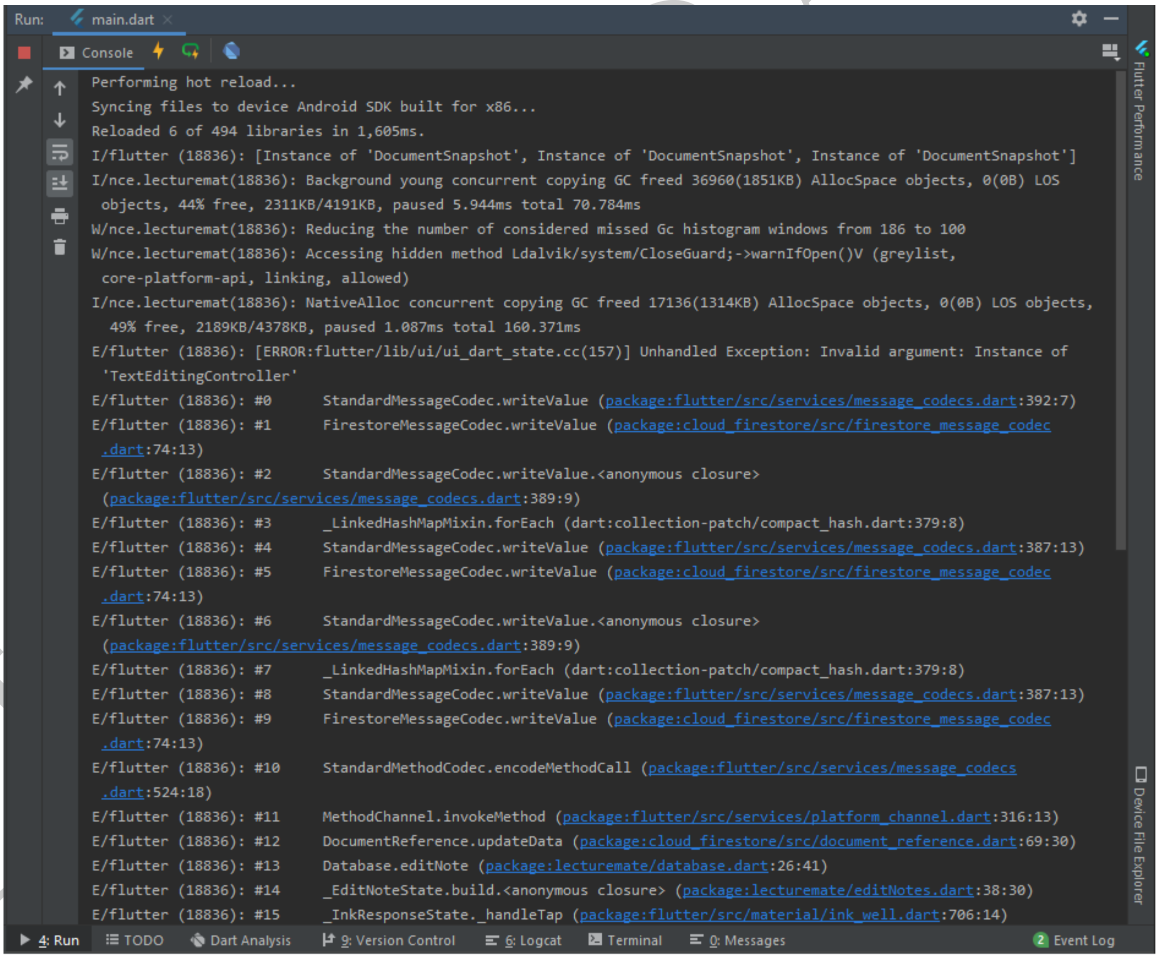
6.6 Disruptions and Solutions



```
═══════ Exception caught by gesture ═══════════════════════════════════════════════════════════════
The following assertion was thrown while handling a gesture:
type 'LabeledGlobalKey<ScaffoldState>' is not a subtype of type 'BuildContext'


Either the assertion indicates an error in the framework itself, or we should provide substantially more information
 in this error message to help you determine and fix the underlying cause.
In either case, please report this assertion by filing a bug on GitHub:
  https://github.com/flutter/flutter/issues/new?template=BUG.md
```

Figure 6.5: Bug 1: Issue with Scaffold - Displaying the structure of the application



```
Performing hot reload...
Syncing files to device Android SDK built for x86...
Reloaded 6 of 494 libraries in 1,605ms.
I/flutter (18836): [Instance of 'DocumentSnapshot', Instance of 'DocumentSnapshot', Instance of 'DocumentSnapshot']
I/nce.lecturemat(18836): Background young concurrent copying GC freed 36960(1851KB) AllocSpace objects, 0(0B) LOS
 objects, 44% free, 2311KB/4191KB, paused 5.944ms total 70.784ms
W/nce.lecturemat(18836): Reducing the number of considered missed Gc histogram windows from 186 to 100
W/nce.lecturemat(18836): Accessing hidden method Ldalvik/system/CloseGuard;->warnIfOpen()V (greylist,
 core-platform-api, linking, allowed)
I/nce.lecturemat(18836): NativeAlloc concurrent copying GC freed 17136(1314KB) AllocSpace objects, 0(0B) LOS objects,
  49% free, 2189KB/4378KB, paused 1.087ms total 160.371ms
E/flutter (18836): [ERROR:flutter/lib/ui/ui_dart_state.cc(157)] Unhandled Exception: Invalid argument: Instance of
 'TextEditingController'
E/flutter (18836): #0      StandardMessageCodec.writeValue (package:flutter/src/services/message_codecs.dart:392:7)
E/flutter (18836): #1      FirestoreMessageCodec.writeValue (package:cloud_firestore/src/firestore_message_codec
 .dart:74:13)
E/flutter (18836): #2      StandardMessageCodec.writeValue.<anonymous closure>
 (package:flutter/src/services/message_codecs.dart:389:9)
E/flutter (18836): #3      _LinkedHashMapMixin.forEach (dart:collection-patch/compact_hash.dart:379:8)
E/flutter (18836): #4      StandardMessageCodec.writeValue (package:flutter/src/services/message_codecs.dart:387:13)
E/flutter (18836): #5      FirestoreMessageCodec.writeValue (package:cloud_firestore/src/firestore_message_codec
 .dart:74:13)
E/flutter (18836): #6      StandardMessageCodec.writeValue.<anonymous closure>
 (package:flutter/src/services/message_codecs.dart:389:9)
E/flutter (18836): #7      _LinkedHashMapMixin.forEach (dart:collection-patch/compact_hash.dart:379:8)
E/flutter (18836): #8      StandardMessageCodec.writeValue (package:flutter/src/services/message_codecs.dart:387:13)
E/flutter (18836): #9      FirestoreMessageCodec.writeValue (package:cloud_firestore/src/firestore_message_codec
 .dart:74:13)
E/flutter (18836): #10     StandardMethodCodec.encodeMethodCall (package:flutter/src/services/message_codecs
 .dart:524:18)
E/flutter (18836): #11     MethodChannel.invokeMethod (package:flutter/src/services/platform_channel.dart:316:13)
E/flutter (18836): #12     DocumentReference.updateData (package:cloud_firestore/src/document_reference.dart:69:30)
E/flutter (18836): #13     Database.editNote (package:lecturemate/database.dart:26:41)
E/flutter (18836): #14     _EditNoteState.build.<anonymous closure> (package:lecturemate/editNotes.dart:38:30)
E/flutter (18836): #15     _InkResponseState._handleTap (package:flutter/src/material/ink_well.dart:706:14)
```

Figure 6.6: Bug 2: Issue with retrieving data from Firebase

Chapter 7

Testing

7.1 Testing Criteria

7.2 Testing Results

7.3 Formative Evalutation

7.4 Functional Requirements Evaluation

7.5 Non-functional Requirements Evaluation

Chapter 8

Conclusion

8.1 Summative Evaluation

8.2 Potential Future Development

Chapter 9

References

Chapter 10

Appendix

END OF DOCUMENT