*An mini project report on*

Digital Institute

Submitted By

**RAJAN MANI TRIPATHI(Regd. No: 17W91A05B0)**

**K. SUMANTH REDDY (Regd. No: 17W91A0588)**

**MOHD SULTAN RAYNI(Regd. No: 17W91A0595)**

Under the esteemed guidance of

**Mrs. K.SAI MADHURI**

Assistant Professor, CSE

*To*

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD**In partial fulfillment of the requirements for award of degree of

****

**BACHELOR OF TECHNOLOGY  
IN  
COMPUTER SCIENCE AND ENGINEERING**

**2017**– **2021**

**MALLA REDDY  
INSTITUTE OF ENGINEERING & TECHNOLOGY, (MRET - W9)** (Sponsored by Malla Reddy Educational society)

Permanently affiliated to JNTUH, Approved by AICTE, Accredited by NBA & NAAC, An ISO 9001:2015 Certified Institution Maisammaguda, Dhulapally post, Malkajgiri, Medchal-500100.

**DECLARATION**

We hereby declare that the project entitled “DIGITAL INSTITUTE ” submitted to Malla Reddy Institute of Engineering and Technology (MRET-W9), affiliated to Jawaharlal Nehru Technological University Hyderabad (JNTUH) for the award of the degree of Bachelor of Technology in Computer Science & Engineering is a result of original industrial oriented project done by us.

It is further declared that the project report or any part thereof has not been previously submitted to any University or Institute for the award of degree or diploma.

**RAJAN MANI TRIPATHI (Regd. No: 17W91A05B0)**

**K. SUMANTH REDDY (Regd. No: 17W91A0588)**

**MOHD SULTAN RAYNI(Regd. No: 17W91A0595)**

MALLA REDDY INSTITUTE OF ENGINEERING & TECHNOLOGY

Sponsored by Malla Reddy Educational Society) Affiliated to JNTU, Hyderabad Maisammaguda, Dhulapally (post via Hakimpet), Sec’Bad-500 014.Phone: 040-65969674, Cell: 9348161223

Department of Computer Science and Engineering

BONAFIED CERTIFICATE

This is to certify that this is the Bonafide Certificate of the Project Titled “**DIGITAL INSTITUTE** ” is submitted by **RAJAN MANI TRIPATHI (Regd. No: 17W91A05B0), K. SUMANTH REDDY (Regd. No: 17W91A0588)** and **MOHD SULTAN RAYNI(Regd. No: 17W91A0595)** of B.Tech in the partial fulfillment of the requirements for the degree of **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING**, Department of Computer Science & Engineering and this has not been submitted for the award of any other degree of this institution.

VIVA VOCE DATE :.................................

**INTERNAL GUIDE HEAD OF THE DEPARTMENT**

**EXTERNAL EXAMINER**

**ACKNOWLEDGEMENT**

First and foremost, we are grateful to the Principal **Dr. M ASHOK** Professor, for providing us with all the resources in the college to make our project a success. I thank him for his valuable suggestions at the time of seminars which encouraged us to give our best in the project.

We would like to express our gratitude to **Dr. P KIRAN KUMAR REDDY** Professor, Dean of Academics for his support and valuable suggestions during the dissertation work.

We would like to express our gratitude to **Dr. ANANTHA RAMAN G R** Professor**,** Head of the Department and our internal guide, Department of Computer Science and Engineering for his support and valuable suggestions during the dissertation work

We offer our sincere gratitude to our project -coordinator **Mr. G NAGAPPA** Assistant Professor of Computer Science and Engineering department who has supported us throughout this project with their patience and valuable suggestions.

We would also like to thank all the supporting staff of the Dept. of CSE and all other departments who have been helpful directly or indirectly in making the project a success.

We are extremely grateful to our parents for their blessings and prayers for our completion of project that gave us strength to do our project.

**RAJAN MANI TRIPATHI**

**K. SUMANTH REDDY**

**MOHD SULTAN RAYNI**

**INDEX**

Abstract I  
List of Figures II  
List of Screens III  
List of Tables III  
List of Abbreviations IV

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CH.**  **NO.** | **CONTENT** | | | **PAGE NO.** |
| **1** | **INTRODUCTION** | | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
| **2** | **LITERATURE SURVEY** | | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  | |  |
| **3** | **SYSTEM ANALYSIS** | | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CH.**  **NO.** | **CONTENT** | | | **PAGE NO.** |
|  |  |  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  | |  |
| **4** | **SYSTEM DESIGN** | | | **24-29** |
|  |  | |  |
|  |  | |  |
|  |  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  | |  |
| **5** | **TECHNOLOGIES USED** | | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
| **6** | **SYSTEM ENVIRONMENT SETUP** | | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
| **7** | **SYSTEM IMPLEMENTATION** | | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |
|  |  | |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **CH.**  **NO.** | **CONTENT** | | **PAGE NO.** |
|  |  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| **8** | **SYSTEM TEST** | |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| **9** | **CONCLUSION** | |  |
|  |  |  |
|  |  |  |
|  | **REFERENCES** | |  |

**ABSTRACT**

The main idea behind the application is making a platform for student and faculty intercation avaible for all time and a secure private channel for research and lecuture . This application is developed to create a private institution specific platform between student and faculty such that they can subimt their ideas by means for pdf, give explantion by means of video and communticat with the institution research department at any time any where. Its’s an intercative andrioid aplication availabe for per perosn and complete control over the application is provided by means of firebase.The application also help to view the lectures if the student miss the class by some reason.The library can be maintained for easy access of the student and faculty by means of pdf formated books,newletters,research paper uploaded in the application server.

**LIST OF FIGURES**

**LIST OF SCREENS**

**1. INTRODUCTION**

**INTRODUCTION**

Digital Institute was designed as a practical approach to the problem of interface between students and lectures. Digital Institute is an android application which can be used to upload recorded video,written notes,books pdf,communication interface. This project aims to provide a simple interactive and intuitive approach to view videos and library uploaded by faculty and students.

**PROBLEM STATEMENT**

To Create an Application for a Institute so that they can privatize their research,classes and library .

**OBJECTIVES**

The main objective of DIGITAL INSTITUTE is to provide users with a simple interactive approach of viewing video classes,accessing e-library,uploading notes ,etc...

* Provides users with a simple interface to interact with.Institute
* Institute privatize network
* Interface for online classes and lectures
* Create a responsive system.
* Create a library and view access

**SCOPE**

The scope of this project is mainly in research and institutions that like their data should be privatized to them.The institute which like to have their own library and video streaming and recording in their resources.

**LIMITATIONS**

* DIGITAL INSTITUTE requires a continuous internet connection.
* DIGITAL INSTITUTE is dependent on firebase server**.**
* DIGITAL INSTITUTE currently doesn’t store Users history for Recommendations,or video anlysis
* DIGITAL INSTITUTE Currenly doesn’t have communication facilities.

**CONCLUSION**

This section provides insight into the project, the various limitations of the project and the various advantages of it. Here we get a clear picture of what the system must do and what is expected of the system.

**2. LITERATURE SURVEY**

**INTRODUCTION**

**A literature survey or a literature review in a project report is that section which shows the various analyses and research made in the field of your interest and the results already published, considering the various parameters of the project and the extent of the project.**

It is the most important part of your report as it gives you a direction in your research. It helps you set a goal for your analysis - thus giving you your problem statement.

**EXISTING SYSTEMS**

The field of DIGITAL INSTITUTE is a weakly explored field with limited approaches .Here We will discuss about some of the Approches.

E-Library:

E-libray is proposed to facilitate providing just books pdf. It’s provide the books collection by university but their is no feature for university to manage or give facilties to lectures and students to upload their work in the platform.Only the recommend admin has access to upload or change the server data.

**DISADVANTAGES OF EXISTING SYSTEM**

**The existing systems have the following disadvantages:**

1. Only supports pdf upload by specified users.
2. It is Expensive for implementation
3. It Contains Complicated Interfaces

**PROPOSED SYSTEM**

The Proposed System is one of the best cost efficient approach for DIGITAL INSTITUTE.It doesn’t depends only by specfic users to upload the few books and videos in server.Its help institute to have a private network for them.

**CONCLUSION**

From the literature survey we can conclude that we have overcome the drawbacks of existing system and provide a clean and easy to system for the users. We have studied the various problems with the current system and have come up with a system to overcome these problems.

**3.SYSTEM ANALYSIS**

**INTRODUCTION**

A software requirements specification (SRS) is a document that captures complete description about how the system is expected to perform. It is usually signed off at the end of requirements engineering phase. It is a description of a software system to be developed. It lays out functional and non-functional requirements and may include a set of use cases that describe user interactions that the software must provide.

The important parts of the Software Requirements Specification (SRS) document are:

⊗Functional requirements of the system

⊗Non-functional requirements of the system

⊗Goals of implementation

These are the important parts of an SRS.

**FUNCTIONAL REQUIREMENTS OF THE SYSTEM**

The main functional requirements of DIGITAL INSTITUTE are:

⊗It must provide an interface for users to log into the system.  
⊗It must provide new users an interface to register into the system.  
⊗It should automatically distinguish between a normal user and an admin.  
⊗The system must provide the users with an interface for uploading new videos and books.  
⊗The system must provide users with an interface to view their uploaded videos and books.

⊗The system must provide users with an interface to view the status of their uploaded videos and books.

⊗Admin must be able to remove and add user from server too..  
⊗All users must be able to view all approves videos and books.

**NON-FUNCTIONAL REQUIREMENTS OF THE SYSTEM**

The non-functional requirements of the system are:

⊗ACCESS SECURITY - The extent to which the system is safeguarded against deliberate and intrusive faults from internal and external sources

⊗AVAILABILITY - The degree to which users can depend on the system to be up (able to function) during “normal operating times”

⊗CONFIDENTIALITY - The degree to which the software system protects sensitive data and allows only authorized access to the data

⊗RELIABILITY - The extent to which the software system consistently performs the specified functions without failure

⊗USABILITY - The ease with which the user can learn, operate, prepare inputs, and interpret outputs through interaction with a system.

⊗INTEGRITY - The degree to which the data maintained by the software system are accurate, authentic, and without corruption.

**GOALS OF IMPLEMENTATION**

The main goal of this project is to develop a simple system that is user friendly and easily understandable. This system aims to provide a practical approach to do Digital Institute.

**SYSTEM SPECIFICATIONS**

**Frontend**

**JAVA**

Java is a high-level programming language developed by Sun Microsystems. It was originally designed for developing programs for set-top boxes and handheld devices, but later became a popular choice for creating web applications and mobile Applications . The Java syntax is similar to [C++](https://techterms.com/definition/cplusplus), but is strictly an Object Oriented Program.

**XML**

Extensible Mark-up Language (XML) is used to describe data. The XML standard is a flexible way to create information formats and electronically share structured data via the public Internet as well as via corporate networks. It is Used to Create GUI for the Android Studio.

**Backend**

**JAVA**

Java is a high-level programming language developed by Sun Microsystems. It was originally designed for developing programs for set-top boxes and handheld devices, but later became a popular choice for creating web applications and mobile Applications . The Java syntax is similar to [C++](https://techterms.com/definition/cplusplus), but is strictly an Object Oriented Program.

**Firebase**

**Firebase** is a Backend-as-a-Service (Baas). It provides developers with a variety of tools and services to help them develop quality apps, grow their user base, and earn profit. It is built on Google’s infrastructure.

**Firebase Authentication**

It supports authentication using passwords, phone numbers, Google, Facebook, Twitter, and more. The Firebase Authentication (SDK) can be used to manually integrate one or more sign-in methods into an app.

### Realtime database

Data is synced across all clients in realtime and remains available even when an app goes offline.

**Development Software**

**Android Studio**

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA . On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

* A flexible Gradle-based build system
* A fast and feature-rich emulator
* A unified environment where you can develop for all Android devices
* Apply Changes to push code and resource changes to your running app without restarting your app
* Code templates and GitHub integration to help you build common app features and import sample code
* Extensive testing tools and frameworks
* Lint tools to catch performance, usability, version compatibility, and other problems
* C++ and NDK support
* Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine

**CONCLUSION**

The analysis tells as the requirement specifications of the project. The functional requirements specify the functionality and functional requirements were as the software requirements tell the required software and supporting files to process the data. The hardware requirements tell about the hardware components required to run the software. The various requirements of the system are selected through rigorous survey, the development is done in such a way that we ensure that all the requirements are met, and the software is up to the standards of a professional software.

**4.SYSTEM DESIGN**

**INTRODUCTION**

System Design is the process or art of defining the architecture components, modules, interfaces and data for a system to satisfy specified requirements. One should see as the applications of the systems theory to product development.

System design is the phase that bridges the gap between the problem domain and the existing system in a manageable way. It is the phase where the SRS document is converted into a format that can be implemented and decides how the system will operate.

In this phase, the complex activity of system development is divided into several smaller sub- activities, which coordinate with each other to achieve the main objective of system development.

**UNIFIED MODELLING LANGUAGE (UML)**

Unified Modelling Language (UML) is a general-purpose modelling language. The main aim of UML is to define a standard way to visualize the way a system has been designed. It is quite like blueprints used in other fields of engineering.

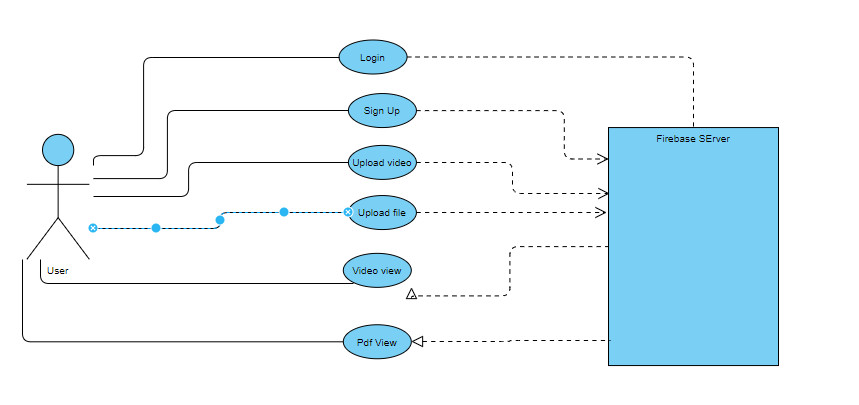
UML is not a programming language; it is rather a visual language. We use UML diagrams to portray the behaviour and structure of a system. UML helps software engineers, businessmen and system architects with modelling, design and analysis. The Object Management Group (OMG) adopted Unified Modelling Language as a standard in 1997. It’s been managed by OMG ever since. International Organization for Standardization (ISO) published UML as an approved standard in 2005. UML has been revised over the years and is reviewed periodically.

UML is linked with object-oriented design and analysis. UML makes the use of elements and forms associations between them to form diagrams. Diagrams in UML can be broadly classified as:

1. **Structural Diagrams** – Capture static aspects or structure of a system. Structural Diagrams include Component Diagrams, Object Diagrams, Class Diagrams and Deployment Diagrams.

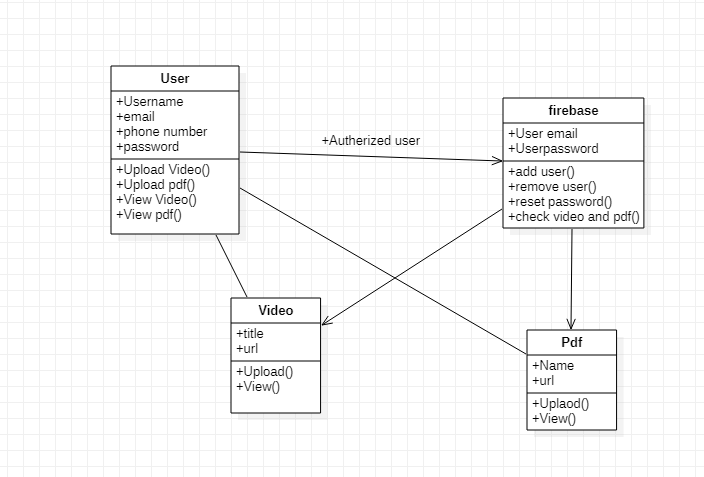
2. **Behaviour Diagrams** – Capture dynamic aspects or behaviour of the system. Behaviour diagrams include Use Case Diagrams, State Diagrams, Activity Diagrams and Interaction Diagrams.

**Use Case Diagrams –** Use Case Diagrams are used to depict the functionality of a system or a part of a system. They are widely used to illustrate the functional requirements of the system and its interaction with external agents(actors). A use case is a diagram representing different scenarios where the system can be used. A use case diagram gives us a high-level view of what the system or a part of the system does without going into implementation details.



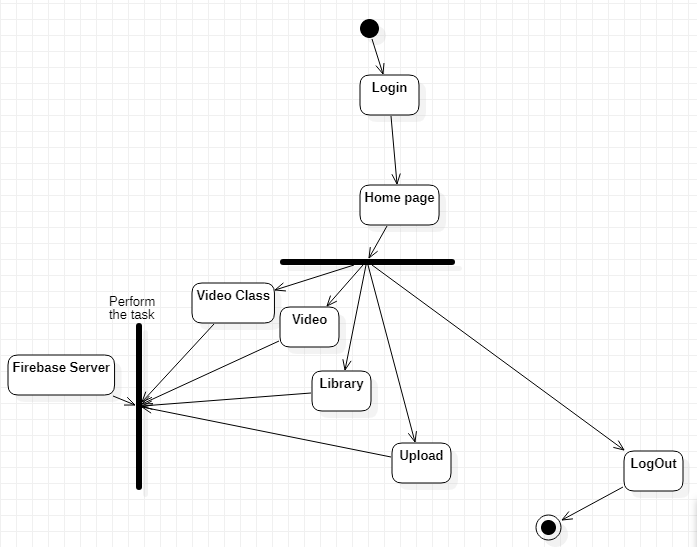
|  |
| --- |
| Fig 4.1 Use Case Diagram |

**Class Diagram –** The most widely use UML diagram is the class diagram. It is the building block of all object-oriented software systems. We use class diagrams to depict the static structure of a system by showing the system’s classes, their methods and attributes. Class diagrams also help us identify the relationship between different classes or objects.



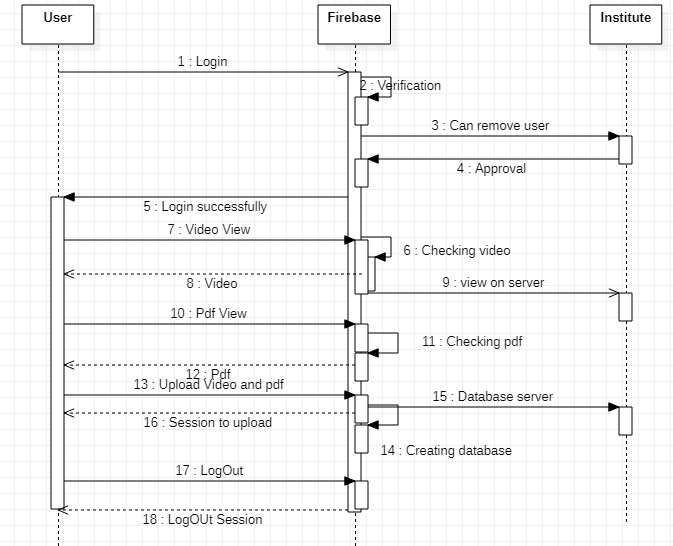
|  |
| --- |
| Fig 4.2 Class Diagram |

**Activity Diagrams –** We use Activity Diagrams to illustrate the flow of control in a system. We can also use an activity diagram to refer to the steps involved in the execution of a use case. We model sequential and concurrent activities using activity diagrams. So, we depict workflows visually using an activity diagram. An activity diagram focuses on the condition of flow and the sequence in which it happens. We describe or depict what causes an event using an activity diagram.



|  |
| --- |
| Fig 4.3 Activity Diagram |

**Sequence Diagram –** A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.



|  |
| --- |
| Fig 4.4 Sequence Diagram |

**5. TECHNOLOGIES USED**

**INTRODUCTION**

Any system requires the implementation of various technologies which together help the system run. These technologies include both hardware and software. The seamless integration between these components helps the system run smoothly. If there is no proper integration between these components the system may develop unwanted complications. This project takes advantage of multiple open-source projects and would like to thank. Some of the technologies used in this project include:

* Java
* XML
* Android
* Firebase
* Exoplayer
* Dexter

**Java**

The Java programming language is...

* **General-purpose**: It is designed to be used for writing software in a wide variety of application domains, and lacks specialized features for any specific domain.
* **Class-based**: Its object structure is defined in classes. Class instances always have those fields and methods specified in their class definitions (see Classes and Objects). This is in contrast to non-class-based languages such as JavaScript.
* **Statically-typed**: the compiler checks at compile time that variable types are respected. For example, if a method expects an argument of type String, that argument must in fact be a string when the method is called.
* **Object-oriented**: most things in a Java program are class instances, i.e. bundles of state (fields) and behavior (methods which operate on data and form the object's interface to the outside world).
* **Portable**: It can be compiled on any platform with javac and the resultant class files can run on any platform that has a JVM.

Java is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation.

Java code is compiled to bytecode (the .class files) which in turn get interpreted by the Java Virtual Machine (JVM). In theory, bytecode created by one Java compiler should run the same way on any JVM, even on a different kind of computer. The JVM might (and in real-world programs will) choose to compile into native machine commands the parts of the bytecode that are executed often. This is called "Just-in-time (JIT) compilation".

# Java Editions and Versions

There are three "editions" of Java defined by Sun / Oracle:

* Java Standard Edition (SE) is the edition that is designed for general use.
* Java Enterprise Edition (EE) adds a range of facilities for building "enterprise grade" services in Java. Java EE is covered separately.
* Java Micro Edition (ME) is based on a subset of Java SE and is intended for use on small devices with limited resources.

There is a separate topic on Java SE / EE / ME editions.

Each edition has multiple versions. The Java SE versions are listed below.

**XML**

XML is a markup language used to store hierarchical data in text files. It is also known as semi-structured data, like JSON. XML is machine-readable, yet can also be read and produced by people.

XML is made up of elements, sometimes casually referred to as a tag soup, which can themselves contain other elements and/or text. Elements may also contain attributes.

XML is often used for data exchange between platforms, especially over the internet. It is also increasingly used for storing semi-structured data in NoSQL data stores (XML databases/document stores). Furthermore, it has the flexibility to handle document-oriented data (text with markup), which makes it very popular in the publishing industry. XML is also widely used for configuration files.

One of the main reasons why XML is so widely used is that it is standardized, with many parsers available, including open source. This makes the cost of using XML lower than the invention of one's own new syntax.

More information about XML's origin and goals can be found in the official [W3C Recommendation](https://www.w3.org/TR/2006/REC-xml-20060816/#sec-origin-goals).

There are two versions of XML, shown in the table below. The editions of each version are just revisions of the original documents and not changes of the standards.

The first version of XML is [1.0](https://www.w3.org/TR/2008/REC-xml-20081126/). XML [1.1](https://www.w3.org/TR/2006/PER-xml11-20060614/) was primary changed due to the Unicode version change from 2.0 to 3.1 and specifies a set of new rules for the use and interpretation of new Unicode characters.

**Android**

**Android** is a [mobile operating system](https://en.wikipedia.org/wiki/Mobile_operating_system) based on a modified version of the [Linux kernel](https://en.wikipedia.org/wiki/Linux_kernel) and other [open source](https://en.wikipedia.org/wiki/Open-source_software) software, designed primarily for [touchscreen](https://en.wikipedia.org/wiki/Touchscreen) mobile devices such as [smartphones](https://en.wikipedia.org/wiki/Smartphone) and [tablets](https://en.wikipedia.org/wiki/Tablet_computer). Android is developed by a consortium of developers known as the [Open Handset Alliance](https://en.wikipedia.org/wiki/Open_Handset_Alliance) and commercially sponsored by [Google](https://en.wikipedia.org/wiki/Google). It was unveiled in November 2007, with the [first commercial Android device](https://en.wikipedia.org/wiki/HTC_Dream) launched in September 2008.

It is [free and open source](https://en.wikipedia.org/wiki/Free_and_open_source) software; its source code is known as Android Open Source Project (AOSP), which is primarily licensed under the [Apache License](https://en.wikipedia.org/wiki/Apache_License). However most Android devices ship with additional [proprietary software](https://en.wikipedia.org/wiki/Proprietary_software) pre-installed,[[10]](https://en.wikipedia.org/wiki/Android_(operating_system)#cite_note-10) most notably [Google Mobile Services](https://en.wikipedia.org/wiki/Google_Mobile_Services) (GMS)[[11]](https://en.wikipedia.org/wiki/Android_(operating_system)#cite_note-11) which includes core apps such as [Google Chrome](https://en.wikipedia.org/wiki/Google_Chrome), the [digital distribution](https://en.wikipedia.org/wiki/Digital_distribution) platform [Google Play](https://en.wikipedia.org/wiki/Google_Play) and associated [Google Play Services](https://en.wikipedia.org/wiki/Google_Play_Services) development platform. About 70 percent of Android smartphones run Google's ecosystem;[[12]](https://en.wikipedia.org/wiki/Android_(operating_system)#cite_note-12) competing Android ecosystems and [forks](https://en.wikipedia.org/wiki/Fork_(software_development)) include [Fire OS](https://en.wikipedia.org/wiki/Fire_OS) (developed by [Amazon](https://en.wikipedia.org/wiki/Amazon_(company))) or [LineageOS](https://en.wikipedia.org/wiki/LineageOS). However the "Android" name and logo are [trademarks](https://en.wikipedia.org/wiki/Trademark) of Google which impose standards to restrict "uncertified" devices outside their ecosystem to use Android branding.[[13]](https://en.wikipedia.org/wiki/Android_(operating_system)#cite_note-13)[[14]](https://en.wikipedia.org/wiki/Android_(operating_system)#cite_note-14)

The source code has been used to develop variants of Android on a range of other electronics, such as [game consoles](https://en.wikipedia.org/wiki/Video_game_console), [digital cameras](https://en.wikipedia.org/wiki/Digital_camera), [portable media players](https://en.wikipedia.org/wiki/Portable_media_player), [PCs](https://en.wikipedia.org/wiki/Personal_computer) and others, each with a specialized user interface. Some well known derivatives include [Android TV](https://en.wikipedia.org/wiki/Android_TV) for televisions and [Wear OS](https://en.wikipedia.org/wiki/Wear_OS) for wearables, both developed by Google. Software packages on Android, which use the [APK](https://en.wikipedia.org/wiki/Android_application_package) format, are generally distributed through proprietary [application stores](https://en.wikipedia.org/wiki/Application_store) like [Google Play Store](https://en.wikipedia.org/wiki/Google_Play_Store), [Samsung Galaxy Store](https://en.wikipedia.org/wiki/Samsung_Galaxy_Store), and [Huawei AppGallery](https://en.wikipedia.org/wiki/Huawei_AppGallery), or open source platforms like [Aptoide](https://en.wikipedia.org/wiki/Aptoide) or [F-Droid](https://en.wikipedia.org/wiki/F-Droid).

Android has been the best-selling OS worldwide on smartphones since 2011 and on tablets since 2013. As of May 2017, it has over two billion [monthly active users](https://en.wikipedia.org/wiki/Monthly_active_users), the largest [installed base](https://en.wikipedia.org/wiki/Installed_base) of any operating system, and as of January 2021, the Google Play Store features over 3 million apps.[[15]](https://en.wikipedia.org/wiki/Android_(operating_system)#cite_note-3_million_apps-15) The current stable version is [Android 11](https://en.wikipedia.org/wiki/Android_11), released on September 8, 2020.

**Firebase**

[Firebase](https://firebase.google.com/) is a mobile and web application platform with tools and infrastructure designed to help developers build high-quality apps.

**Features**

Firebase Cloud Messaging, Firebase Auth, Realtime Database, Firebase Storage, Firebase Hosting, Firebase Test Lab for Android, Firebase Crash Reporting.

**Firebase** is a platform developed by [Google](https://en.wikipedia.org/wiki/Google) for creating [mobile](https://en.wikipedia.org/wiki/Mobile_phone_application) and [web](https://en.wikipedia.org/wiki/Web_application) applications. It was originally an independent company founded in 2011.[[1]](https://en.wikipedia.org/wiki/Firebase#cite_note-1) In 2014, Google acquired the platform[[2]](https://en.wikipedia.org/wiki/Firebase#cite_note-2) and it is now their flagship offering for app development.

## **ExoPlayer**

[ExoPlayer](http://google.github.io/ExoPlayer/) is an open source project that is not part of the Android framework and is distributed separately from the Android SDK. ExoPlayer’s standard audio and video components are built on Android’s MediaCodec API, which was released in Android 4.1 (API level 16). Because ExoPlayer is a library, you can easily take advantage of new features as they become available by updating your app.

ExoPlayer supports features like Dynamic adaptive streaming over HTTP (DASH), SmoothStreaming and Common Encryption, which are not supported by [MediaPlayer](https://developer.android.com/reference/android/media/MediaPlayer). It's designed to be easy to customize and extend.

# Dexter

Dexter is an Android library that simplifies the process of requesting permissions at runtime.

Android Marshmallow includes a new functionality to let users grant or deny permissions when running an app instead of granting them all when installing it. This approach gives the user more control over applications but requires developers to add lots of code to support it.

Dexter frees your permission code from your activities and lets you write that logic anywhere you want.

**CONCLUSION**

Working on this project has exposed to the various new and old technologies present in the market for android development. These technologies have specifically been selected due to their versatility and hoe the easily work with each other in tandem. We have gained a lot of insight about the various technologies present in the market and what are their advantages and disadvantages.

**6. SYSTEM ENVIRONMENT SETUP**

**INTRODUCTION**

In this section we will describe how the development and the deployment setup was done for the system. This can be used to replicate the exact workings of the system.

Android Studio Setup

Guide to Install and Set up Android Studio

Android Studio is the official integrated development environment for Google’s Android operating system, built on JetBrains’ IntelliJ IDEA software and designed specifically for Android development.

Operating system: Windows, macOS, Linux

Stable release: 3.1.3 (June 2018; 1 month ago)

Preview release: 3.2 Beta 5 (July 30, 2018; 1 day ago)

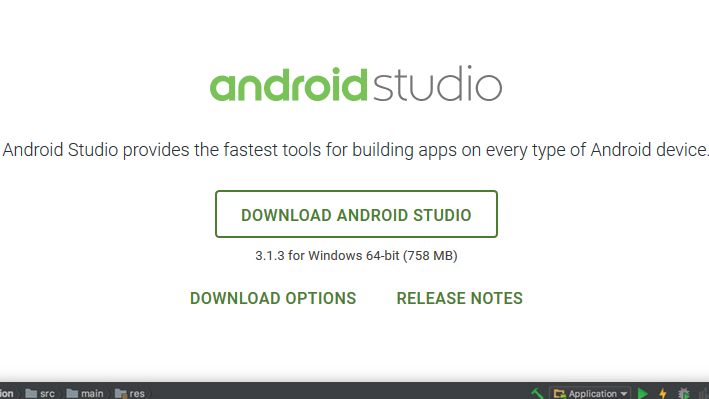
License: Freeware +Source code

Size: 854 MB compressed

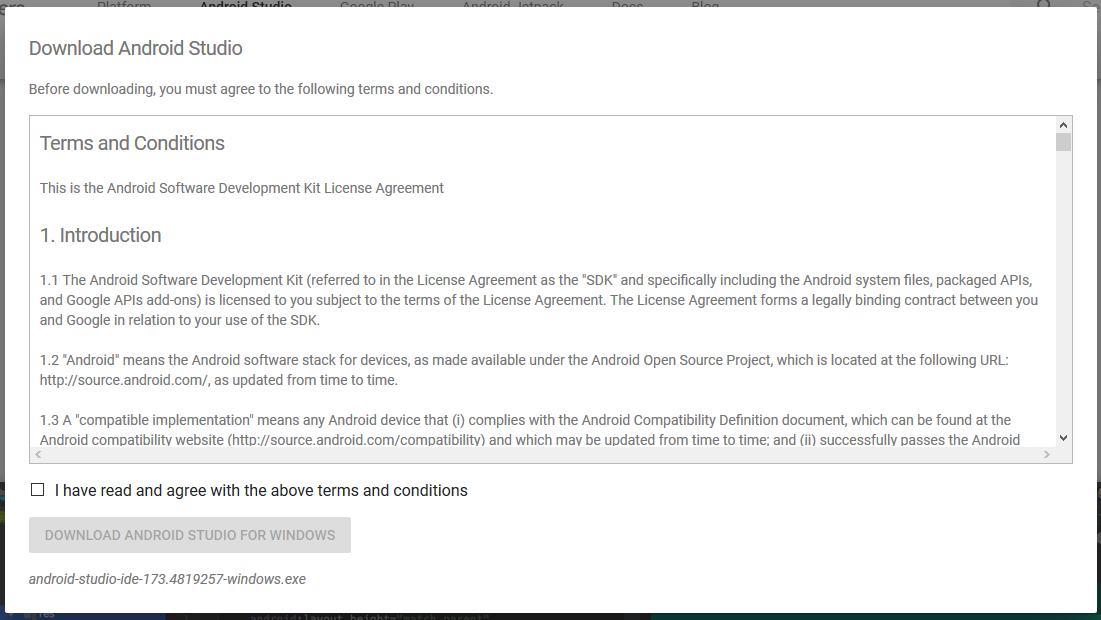
Developed by: Google, JetBrains

**Installation guide:**

* **Step – 1 :**  
  Head over to https://developer.android.com/studio/#downloads to get the Android Studio executable or zip file .
* **Step – 2 :**  
  *Click on the download android studio button .*



*Click on the “I have read and agree with the above terms and conditions” checkbox followed by the download button.*



* *Click on Save file button in the appeared prompt box and the file will start downloading .*
* **Step – 3:**

After the downloading has finished, open the file from downloads and run it .  
It will prompt the following dialogue box .



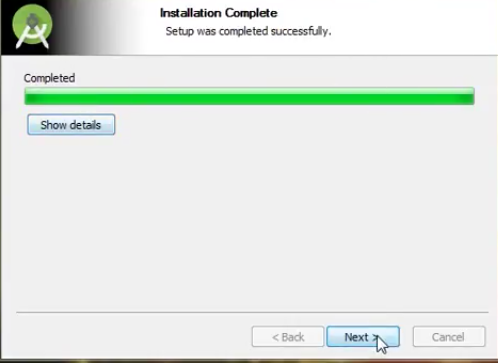
* *Click on next .*

In the next prompt it’ll ask for a path for installation. Choose a path and hit next.

**Note :**The installation path should have the required minimum space.

* **Step – 4 :**

It will start the installation, and once it is completed, it will be like the image shown below .



Click on next *.*



* **Step – 5 :**

Once “Finish” is clicked, it will ask whether the previous settings needs to be impoerted [if android studio had been installed earlier], or not.  
It is better to choose the ‘Don’t import Settings option’ .



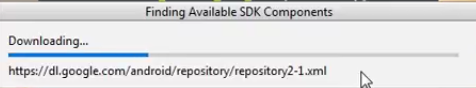
*Click the OK button.*

* **Step – 6 :**

This will start the Android Studio.

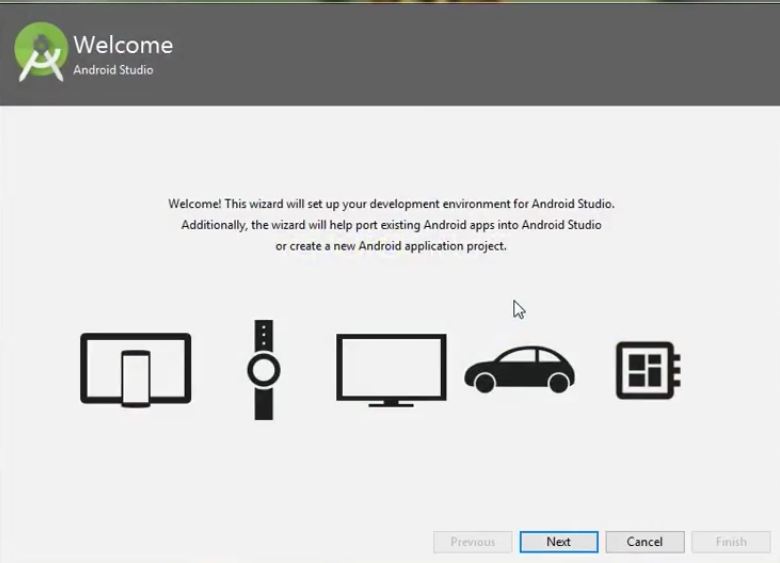


Meanwhile it will be finding the available SDK components .

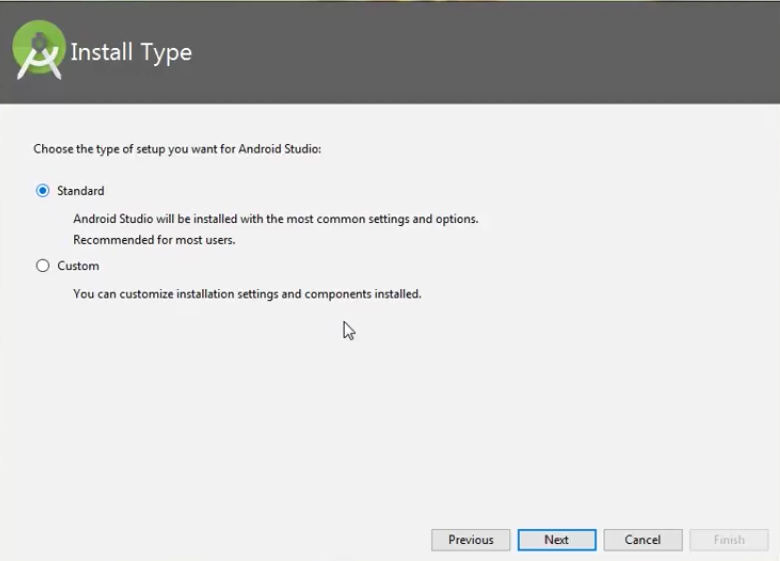


* **Step – 7:**

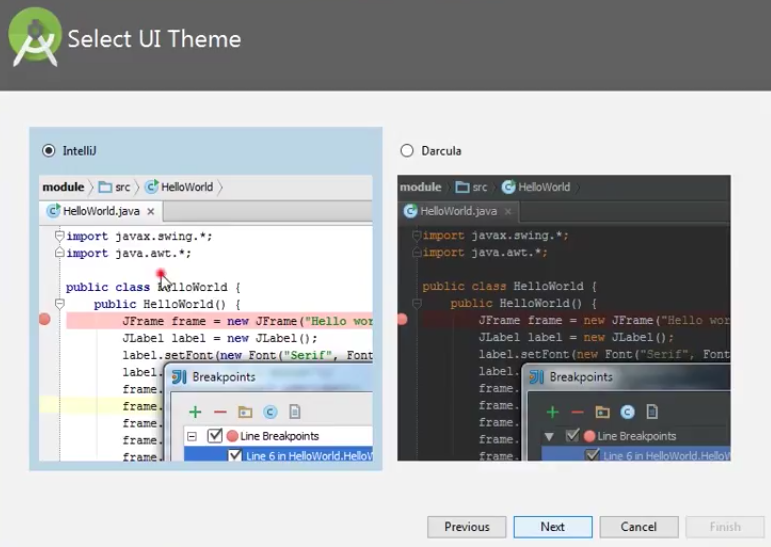
After it has found the SDK components, it will redirect to the Welcome dialog box .



*Click on next .*



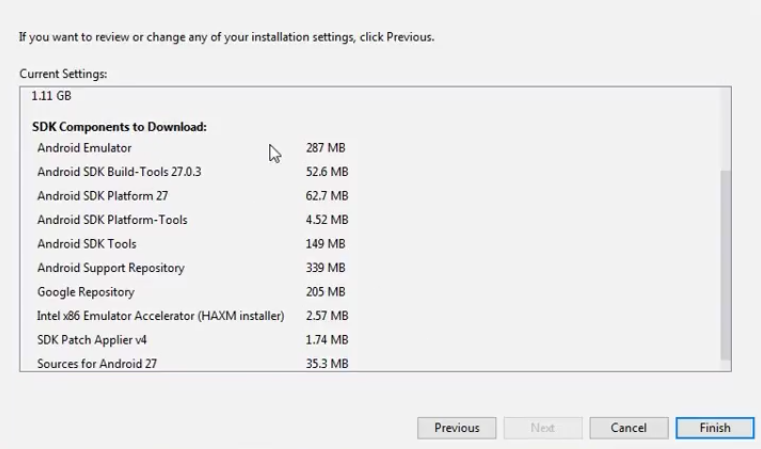
Choose Standard and click on Next.  
Now choose the theme, whether Light theme or the Dark one .  
The light one is called the IntelliJ theme whereas the dark theme is called Darcula .  
Choose as required.



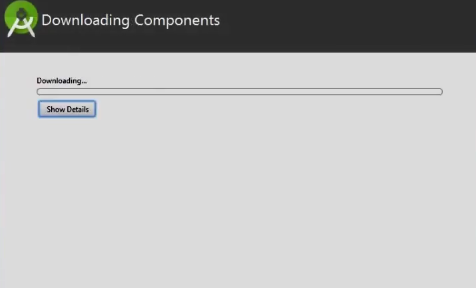
*Click on the Next button*

* **Step – 8 :**

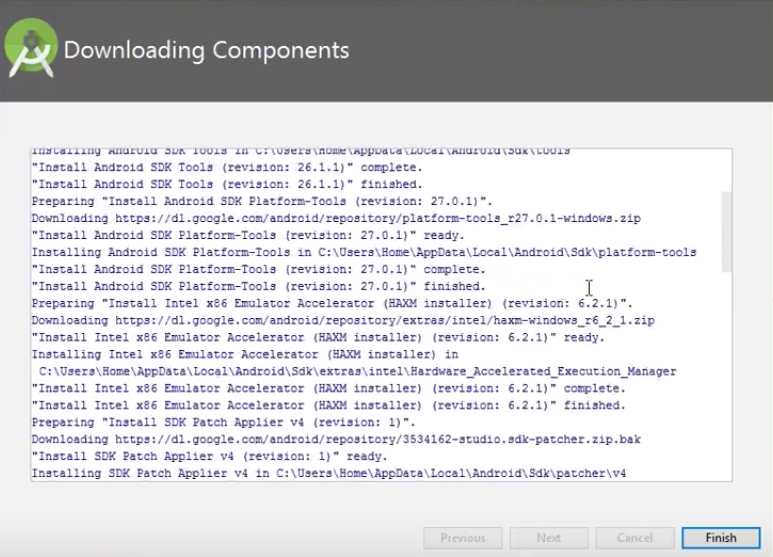
Now it is time to download the SDK components .



*Click on Finish .*



It has started downloading the components

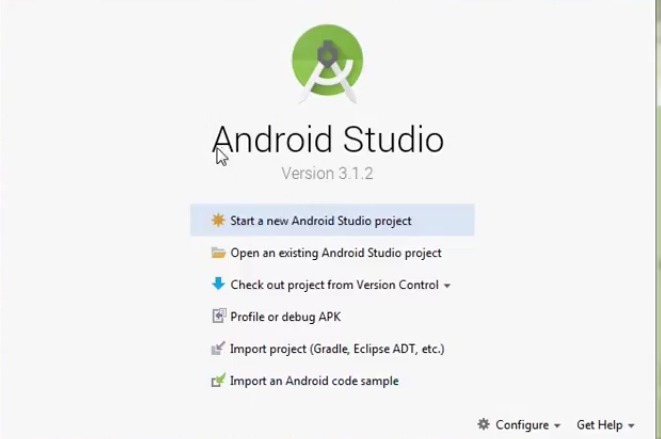


The Android Studio has been successfully configured.  
Now it’s time to launch and build apps.

*Click on the Finish button to launch it.*

* **Step – 9 :**

*Click on ‘Start new android project’ to build a new app.*



**SERVER SETUP**

## Prerequisites

* Install or update [Android Studio](https://developer.android.com/sdk) to its latest version.
* Make sure that your project meets these requirements:
  + Targets API level 16 (Jelly Bean) or later
  + Uses Gradle 4.1 or later
  + Uses [Jetpack (AndroidX)](https://developer.android.com/jetpack/androidx/migrate), which includes meeting these version requirements:
    - com.android.tools.build:gradle v3.2.1 or later
    - compileSdkVersion 28 or later
* Set up a physical device or use an [emulator](https://developer.android.com/studio/run/managing-avds) to run your app.  
  Note that [Firebase SDKs with a dependency on Google Play services](https://firebase.google.com/docs/android/android-play-services) require the device or emulator to have Google Play services installed.
* [Sign into Firebase](https://console.firebase.google.com/) using your Google account.

If you don't already have an Android project and just want to try out a Firebase product, you can download one of our [quickstart samples](https://firebase.google.com/docs/samples).

**You can connect your Android app to Firebase using one of the following options:**

* [**Option 1**](https://firebase.google.com/docs/android/setup#console): (recommended) Use the Firebase console setup workflow.
* [**Option 2**](https://firebase.google.com/docs/android/setup#assistant): Use the Android Studio Firebase Assistant (may require additional configuration).

## Option 1: Add Firebase using the Firebase console

Adding Firebase to your app involves tasks both in the [Firebase console](https://console.firebase.google.com/) and in your open Android project (for example, you download Firebase config files from the console, then move them into your Android project).

### Step 1: Create a Firebase project

Before you can add Firebase to your Android app, you need to create a Firebase project to connect to your Android app. Visit [Understand Firebase Projects](https://firebase.google.com/docs/projects/learn-more) to learn more about Firebase projects.

**Create a Firebase project**

### Step 2: Register your app with Firebase

To use Firebase in your Android app, you need to register your app with your Firebase project. Registering your app is often called "adding" your app to your project.

**Note:** Visit [Understand Firebase Projects](https://firebase.google.com/docs/projects/learn-more#best-practices) to learn more about best practices and considerations for adding apps to a Firebase project, including how to handle multiple build variants.

1. Go to the [Firebase console](https://console.firebase.google.com/).
2. In the center of the project overview page, click the **Android** icon (plat\_android) or **Add app** to launch the setup workflow.
3. Enter your app's package name in the **Android package name** field.

What's a package name, and where do you find it?

Make sure to enter the package name that your app is actually using. The package name value is case-sensitive, and it cannot be changed for this Firebase Android app after it's registered with your Firebase project.

1. (Optional) Enter other app information: **App nickname** and **Debug signing certificate SHA-1**.

How are the *App nickname* and the *Debug signing certificate SHA-1* used within Firebase?

1. Click **Register app**.

### Step 3: Add a Firebase configuration file

1. Add the Firebase Android configuration file to your app:
   1. Click **Download google-services.json** to obtain your Firebase Android config file (google-services.json).
   2. Move your config file into the module (app-level) directory of your app.

What do you need to know about this config file?

1. To enable Firebase products in your app, add the [google-services plugin](https://developers.google.com/android/guides/google-services-plugin) to your Gradle files.
   1. In your root-level (project-level) Gradle file (build.gradle), add rules to include the Google Services Gradle plugin. Check that you have Google's Maven repository, as well.

buildscript {  
  
  repositories {  
    **// Check that you have the following line (if not, add it):  
    google()  // Google's Maven repository**  
  }  
  
  dependencies {  
    // ...  
  
    **// Add the following line:  
    classpath 'com.google.gms:google-services:4.3.5'  // Google Services plugin**  
  }  
}  
  
allprojects {  
  // ...  
  
  repositories {  
    **// Check that you have the following line (if not, add it):  
    google()  // Google's Maven repository**  
    // ...  
  }  
}

* 1. In your module (app-level) Gradle file (usually app/build.gradle), apply the Google Services Gradle plugin:

apply plugin: 'com.android.application'  
**// Add the following line:  
apply plugin: 'com.google.gms.google-services'  // Google Services plugin**  
  
android {  
  // ...  
}

### Step 4: Add Firebase SDKs to your app

1. Using the [Firebase Android BoM](https://firebase.google.com/docs/android/learn-more#bom), declare the dependencies for the [Firebase products](https://firebase.google.com/docs/android/setup#available-libraries) that you want to use in your app. Declare them in your **module (app-level) Gradle file** (usually app/build.gradle).

[Analytics enabled](https://firebase.google.com/docs/web/setup#analytics-enabled-android)[Analytics not enabled](https://firebase.google.com/docs/web/setup#analytics-not-enabled-android)

[Java](https://firebase.google.com/docs/web/setup#java-android)[Kotlin+KTX](https://firebase.google.com/docs/web/setup#kotlin+ktx-android)

dependencies {  
  // ...  
  
  **// Import the Firebase BoM  
  implementation platform('com.google.firebase:firebase-bom:26.4.0')**  
  
  // When using the BoM, you don't specify versions in Firebase library dependencies  
  
  **// Declare the dependency for the Firebase SDK for Google Analytics**  
  **implementation 'com.google.firebase:firebase-analytics'**  
  
  **// Declare the dependencies for any other desired** [**Firebase products**](https://firebase.google.com/docs/android/setup#available-libraries)  
  // For example, declare the dependencies for Firebase Authentication and Cloud Firestore  
  implementation 'com.google.firebase:firebase-auth'  
  implementation 'com.google.firebase:firebase-firestore'  
}

By using the [Firebase Android BoM](https://firebase.google.com/docs/android/learn-more#bom), your app will always use compatible versions of the Firebase Android libraries.

1. Sync your app to ensure that all dependencies have the necessary versions.

That's it! You can skip ahead to check out the recommended [next steps](https://firebase.google.com/docs/android/setup#next_steps).

**CONCLUSION**

In this section we have gone through the environment setup used during development and deployment of the project. We have gained a lot of knowledge on how to practically implement these technologies.

**7. SYSTEM IMPLEMENTATION**

**INTRODUCTION**

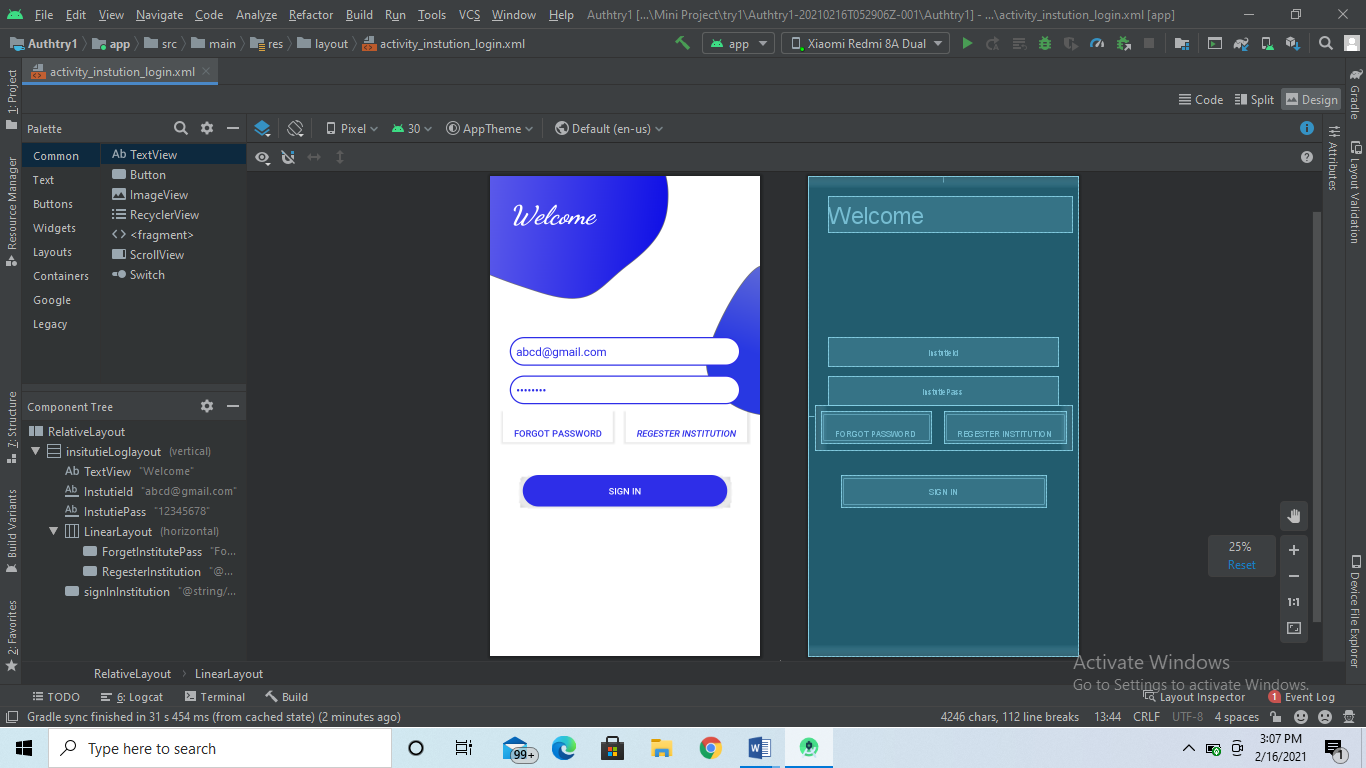
The implementation phase, the project plan is put into motion and the work of the project is performed. The project takes shape during the implementation phase. This phase involves the construction of the actual project result. Programmers are occupied with encoding, designers are involved in developing graphic material, contractors are building, the actual reorganisation takes place.

SYSTEM MODULES

**INSTITUTE LOGIN ACTIVITY**

The login module is the first UI element that the user interacts with. It is a simple interface where the user provides his credentials which include the username and the password. On submission of these, an API request is sent to the Firebase server. On successful verification, the server returns his Account Uid otherwise Shows an Error message. This module is built using JAVA,XML.

XML



**Java**

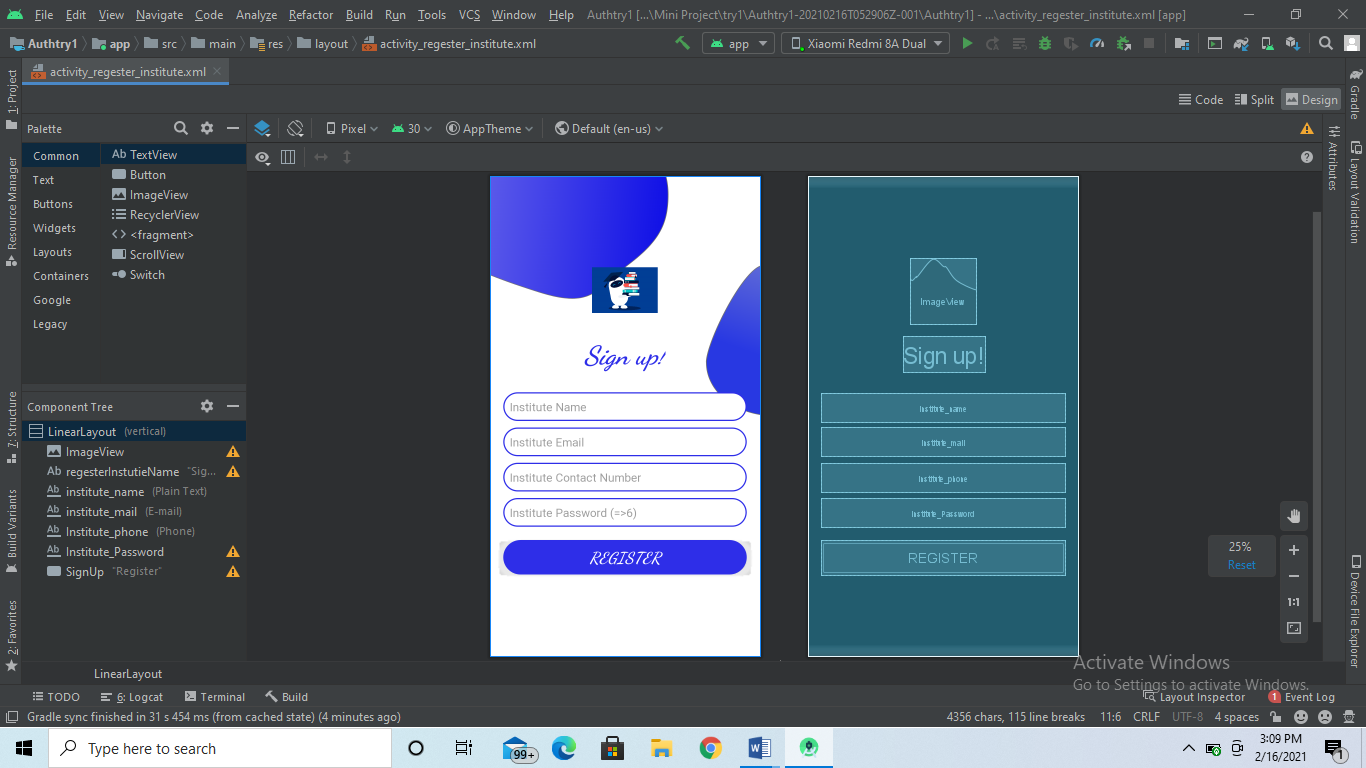
package com.example.authtry1;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.text.TextUtils;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.Toast;  
  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
  
import com.google.android.gms.tasks.OnCompleteListener;  
import com.google.android.gms.tasks.Task;  
import com.google.firebase.auth.AuthResult;  
import com.google.firebase.auth.FirebaseAuth;  
import com.google.firebase.auth.FirebaseUser;  
  
public class InstutionLoginActivity extends AppCompatActivity {  
 EditText InstutionMail, InstutionPass;  
 Button InstutionLogin, InstutionRegester, ForgetInstutionpass;  
  
 FirebaseAuth auth;  
 FirebaseUser user;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_instution\_login*);  
 InstutionMail = findViewById(R.id.*InstutieId*);  
 InstutionPass = findViewById(R.id.*InstutiePass*);  
 InstutionLogin = findViewById(R.id.*signInInstitution*);  
 ForgetInstutionpass = findViewById(R.id.*ForgetInstitutePass*);  
 InstutionRegester = findViewById(R.id.*RegesterInstitution*);  
  
  
 auth = FirebaseAuth.*getInstance*();  
 user = auth.getCurrentUser();  
 if(user!=null)  
 {  
 Intent i = new Intent(InstutionLoginActivity.this, MainActivity.class);  
Intent.FLAG\_ACTIVITY\_NEW\_TASK); check it now  
 startActivity(i);  
 finish();  
  
  
 }  
  
 InstutionRegester.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent i = new Intent(InstutionLoginActivity.this, RegesterInstituteActivity.class);  
  
 startActivity(i);  
 }  
 });  
 ForgetInstutionpass.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent i = new Intent(InstutionLoginActivity.this, ForgetPassword.class);  
 startActivity(i);  
 finish();  
 }  
 });  
 InstutionLogin.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 String InsEmail = InstutionMail.getText().toString();  
 String InsPass = InstutionPass.getText().toString();  
 if (TextUtils.*isEmpty*(InsEmail)) {  
 Toast.*makeText*(InstutionLoginActivity.this, "Please enter the email", Toast.*LENGTH\_SHORT*).show();  
 } else if (TextUtils.*isEmpty*(InsPass)) {  
 Toast.*makeText*(InstutionLoginActivity.this, "Please enter the Password", Toast.*LENGTH\_SHORT*).show();  
  
 } else {  
 auth.signInWithEmailAndPassword(InsEmail, InsPass)  
 .addOnCompleteListener(new OnCompleteListener<AuthResult>() {  
 @Override  
 public void onComplete(@NonNull Task<AuthResult> task) {  
 Toast.*makeText*(InstutionLoginActivity.this, "hello", Toast.*LENGTH\_SHORT*).show();  
 if (task.isSuccessful()) {  
 Intent i = new Intent(InstutionLoginActivity.this, MainActivity.class);  
 i.addFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TASK* | Intent.*FLAG\_ACTIVITY\_NEW\_TASK*);  
 startActivity(i);  
 finish();  
 } else {  
 Toast.*makeText*(InstutionLoginActivity.this, "Institute Login Failed", Toast.*LENGTH\_SHORT*).show();  
 }  
 }  
 });  
 }  
  
  
 }  
 });  
 }  
}

**SIGNUP ACTIVITY**

The Sign Up module is another UI element of the system, it also is built using JAVA,XML. This module is responsible for the registration of the user into the system. This module takes user information like name, email and password. On submission of these details, an API call is made to the Firebase server.

The server first verifies if a user already exists with these credentials, if the user already exists, he will be unable to create an account with that email. If the user does not exist a new entry is made into the database with the user details and a User id is returned to the user which acts as a session identifier.

XML



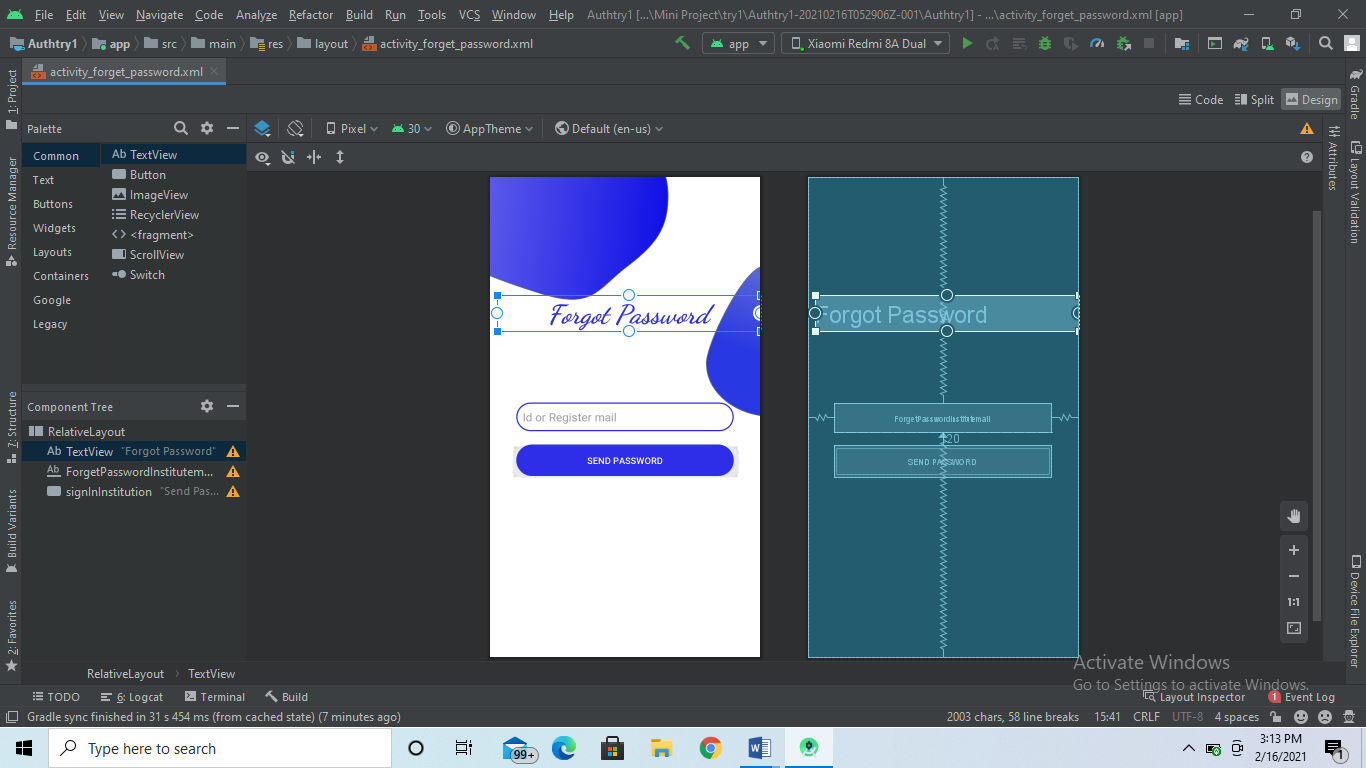
**JAVA**

package com.example.authtry1;  
  
import android.content.Intent;  
import android.net.Uri;  
import android.os.Bundle;  
import android.text.TextUtils;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.Toast;  
  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
  
import com.google.android.gms.tasks.OnCompleteListener;  
import com.google.android.gms.tasks.Task;  
import com.google.firebase.auth.AuthResult;  
import com.google.firebase.auth.FirebaseAuth;  
import com.google.firebase.auth.FirebaseUser;  
import com.google.firebase.database.DatabaseReference;  
import com.google.firebase.database.FirebaseDatabase;  
import com.google.firebase.storage.FirebaseStorage;  
import com.google.firebase.storage.StorageReference;  
  
import java.util.HashMap;  
  
public class RegesterInstituteActivity extends AppCompatActivity {  
 EditText instutiename, instutiepass, instutiemail, instutiePhone;  
 Button instutieRegester;  
 FirebaseAuth auth;  
 DatabaseReference InstutieRef;  
 StorageReference storage;  
 Uri imageuri;  
  
 String userid;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_regester\_institute*);  
// instilaizing edit text view and button  
 instutiename = findViewById(R.id.*institute\_name*);  
 instutiemail = findViewById(R.id.*institute\_mail*);  
 instutiepass = findViewById(R.id.*Institute\_Password*);  
 instutiePhone = findViewById(R.id.*Institute\_phone*);  
 instutieRegester = findViewById(R.id.*SignUp*);  
// firebase authentication  
 auth = FirebaseAuth.*getInstance*();  
 storage = FirebaseStorage.*getInstance*().getReference();  
 instutieRegester.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 CheckRegesterData();  
  
  
 }  
 });  
  
 }  
  
 private void CheckRegesterData() {  
 String InstName = instutiename.getText().toString();  
 String InstMail = instutiemail.getText().toString();  
 String InstPass = instutiepass.getText().toString();  
 String InstPhone = instutiePhone.getText().toString();  
 if (InstName.isEmpty() || InstMail.isEmpty() || InstPass.isEmpty() || InstPhone.isEmpty()) {  
  
 if (TextUtils.*isEmpty*(InstName)) {  
 Toast.*makeText*(this, "Instutie Name is left empty", Toast.*LENGTH\_SHORT*).show();  
 } else if (TextUtils.*isEmpty*(InstMail)) {  
 Toast.*makeText*(this, "Instutie mail is left empty", Toast.*LENGTH\_SHORT*).show();  
 } else if (TextUtils.*isEmpty*(InstPass)) {  
 Toast.*makeText*(this, "Instutie password is left empty", Toast.*LENGTH\_SHORT*).show();  
 } else {  
 Toast.*makeText*(this, "Instutie Contact or phone is left empty", Toast.*LENGTH\_SHORT*).show();  
 }  
 } else {  
 InstituteRegester(InstName, InstMail, InstPhone, InstPass);  
  
// Toast.makeText(this, "Instutie Contact or phone is left empty", Toast.LENGTH\_SHORT).show();  
 }  
 }  
  
 private void InstituteRegester(final String name, String mail, final String phone, String password) {  
 auth.createUserWithEmailAndPassword(mail, password)  
 .addOnCompleteListener(new OnCompleteListener<AuthResult>() {  
 @Override  
 public void onComplete(@NonNull Task<AuthResult> task) {  
 if (task.isSuccessful()) {  
 FirebaseUser user = auth.getCurrentUser();  
 userid = null;  
 if (!(null == user)) userid = user.getUid();  
 InstutieRef = FirebaseDatabase.*getInstance*().getReference("InstutieUser").child(userid);  
// hash map data base  
 HashMap<String, String> hashMap = new HashMap<>();  
 hashMap.put("id", userid);  
 hashMap.put("Institute Name", name);  
 hashMap.put("ImageUrl", "default");  
 hashMap.put("Phone num", phone);  
//opening after succsful regestration  
 InstutieRef.setValue(hashMap).addOnCompleteListener(new OnCompleteListener<Void>() {  
 @Override  
 public void onComplete(@NonNull Task<Void> task) {  
 if (task.isSuccessful()) {  
 Intent i = new Intent(RegesterInstituteActivity.this, MainActivity.class);  
 i.addFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TASK* | Intent.*FLAG\_ACTIVITY\_NEW\_TASK*);  
 startActivity(i);  
 finish();  
 }  
 }  
 });  
  
 }  
 }  
 });  
 }  
  
  
}

**FORGOT ACTIVITY**

The forgot module is another UI element of the system, it also is built using JAVA,XML. This module is responsible for the reseting the password to registered mail id. On submission of these details, an API call is made to the Firebase server.

XML

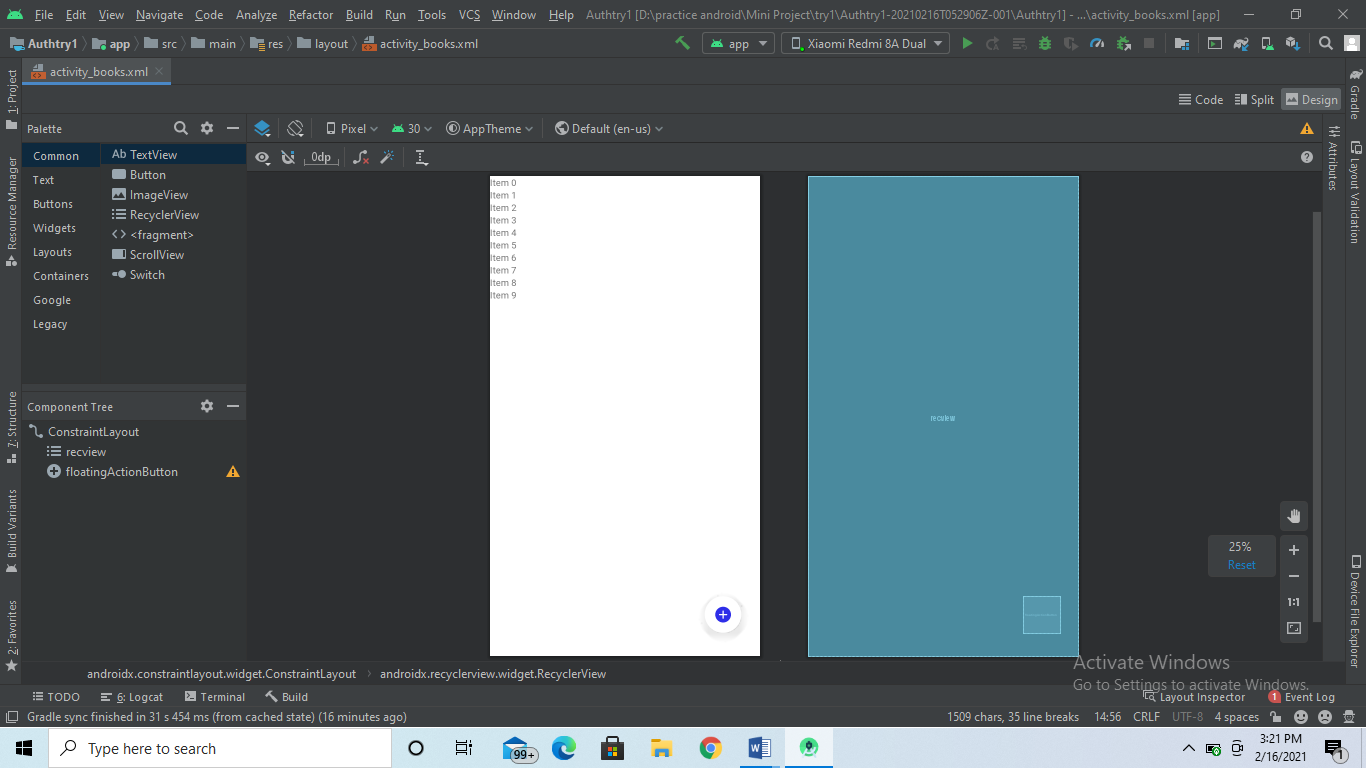
JAVA

package com.example.authtry1;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.EditText;  
import android.widget.Toast;  
  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
  
import com.google.android.gms.tasks.OnCompleteListener;  
import com.google.android.gms.tasks.Task;  
import com.google.firebase.auth.FirebaseAuth;  
  
public class ForgetPassword extends AppCompatActivity {  
 EditText email;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_forget\_password*);  
 email = findViewById(R.id.*ForgetPasswordInstitutemail*);  
  
 }  
  
 public void SendPassword(View view) {  
 FirebaseAuth auth = FirebaseAuth.*getInstance*();  
 String emailAddress = email.getText().toString();  
 auth.sendPasswordResetEmail(emailAddress)  
 .addOnCompleteListener(new OnCompleteListener<Void>() {  
 @Override  
 public void onComplete(@NonNull Task<Void> task) {  
 if (task.isSuccessful()) {  
 Intent i = new Intent(ForgetPassword.this, InstutionLoginActivity.class);  
 i.addFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TASK* | Intent.*FLAG\_ACTIVITY\_NEW\_TASK*);  
 startActivity(i);  
 finish();  
 } else {  
 Toast.*makeText*(ForgetPassword.this, "Password cant be reset or resended", Toast.*LENGTH\_SHORT*).show();  
 }  
 }  
 });  
 }  
}

Books Activity

Use to provide library interface in the applications

XML

Java

package com.example.authtry1;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.view.WindowManager;  
import android.widget.Toast;  
  
import com.firebase.ui.database.FirebaseRecyclerOptions;  
import com.google.android.material.floatingactionbutton.FloatingActionButton;  
import com.google.firebase.database.FirebaseDatabase;  
  
public class BooksActivity extends AppCompatActivity {  
 FloatingActionButton fb;  
 RecyclerView recview;  
 myadapter adapter;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_books*);  
 getWindow().setFlags(WindowManager.LayoutParams.*FLAG\_FULLSCREEN*, WindowManager.LayoutParams.*FLAG\_FULLSCREEN*);  
  
 fb = (FloatingActionButton) findViewById(R.id.*floatingActionButton*);  
 fb.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 startActivity(new Intent(getApplicationContext(), uploadfile.class));  
 }  
 });  
  
 recview = (RecyclerView) findViewById(R.id.*recview*);  
 recview.setLayoutManager(new LinearLayoutManager(this));  
  
 FirebaseRecyclerOptions<model> options =  
 new FirebaseRecyclerOptions.Builder<model>()  
 .setQuery(FirebaseDatabase.*getInstance*().getReference().child("mydocuments"), model.class)  
 .build();  
  
 adapter = new myadapter(options);  
 recview.setAdapter(adapter);  
  
  
 }  
  
 @Override  
 protected void onStart() {  
 super.onStart();  
 adapter.startListening();  
 }  
  
 @Override  
 protected void onStop() {  
 super.onStop();  
 adapter.stopListening();  
 }  
  
}

MyAdapter class used for recycler view listener

package com.example.authtry1;  
  
import android.content.Intent;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.ImageView;  
import android.widget.TextView;  
  
import androidx.annotation.NonNull;  
import androidx.recyclerview.widget.RecyclerView;  
  
import com.firebase.ui.database.FirebaseRecyclerAdapter;  
import com.firebase.ui.database.FirebaseRecyclerOptions;  
  
public class myadapter extends FirebaseRecyclerAdapter<model,myadapter.myviewholder>  
{  
  
 public myadapter(@NonNull FirebaseRecyclerOptions<model> options) {  
 super(options);  
 }  
  
 @Override  
 protected void onBindViewHolder(@NonNull final myviewholder holder, int position, @NonNull final model model) {  
  
 holder.header.setText(model.getFilename());  
  
 holder.textviewbook.setText(String.*valueOf*(model.getNov()));  
 holder.textlike.setText(String.*valueOf*(model.getNol()));  
 holder.textdislike.setText(String.*valueOf*(model.getNod()));  
  
 holder.img1.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 Intent intent=new Intent(holder.img1.getContext(),viewpdf.class);  
 intent.putExtra("filename",model.getFilename());  
 intent.putExtra("fileurl",model.getFileurl());  
  
 intent.setFlags(Intent.*FLAG\_ACTIVITY\_NEW\_TASK*);  
 holder.img1.getContext().startActivity(intent);  
 }  
 });  
  
 }  
  
 @NonNull  
 @Override  
 public myviewholder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
 View view= LayoutInflater.*from*(parent.getContext()).inflate(R.layout.*singelrowdesign*,parent,false);  
 return new myviewholder(view);  
 }  
  
 public class myviewholder extends RecyclerView.ViewHolder  
 {  
 ImageView img1;  
 TextView header;  
 ImageView readbook,likebook,dislikebook;  
 TextView textviewbook,textlike,textdislike;  
  
 public myviewholder(@NonNull View itemView)  
 {  
 super(itemView);  
  
 img1=itemView.findViewById(R.id.*img1*);  
 header=itemView.findViewById(R.id.*header*);  
  
 readbook=itemView.findViewById(R.id.*readbook*);  
 likebook=itemView.findViewById(R.id.*likebook*);  
 dislikebook=itemView.findViewById(R.id.*dislikebook*);  
  
 textviewbook=itemView.findViewById(R.id.*textviewbook*);  
 textlike=itemView.findViewById(R.id.*textlike*);  
 textdislike=itemView.findViewById(R.id.*textdislike*);  
 }  
 }  
}

Model of pdf file

package com.example.authtry1;  
  
public class model  
{  
 String filename, fileurl;  
 int nod,nol,nov;  
  
 public model() {  
 }  
  
 public model(String filename, String fileurl, int nod, int nol, int nov) {  
 this.filename = filename;  
 this.fileurl = fileurl;  
 this.nod = nod;  
 this.nol = nol;  
 this.nov = nov;  
 }  
  
 public String getFilename() {  
 return filename;  
 }  
  
 public void setFilename(String filename) {  
 this.filename = filename;  
 }  
  
 public String getFileurl() {  
 return fileurl;  
 }  
  
 public void setFileurl(String fileurl) {  
 this.fileurl = fileurl;  
 }  
  
 public int getNod() {  
 return nod;  
 }  
  
 public void setNod(int nod) {  
 this.nod = nod;  
 }  
  
 public int getNol() {  
 return nol;  
 }  
  
 public void setNol(int nol) {  
 this.nol = nol;  
 }  
  
 public int getNov() {  
 return nov;  
 }  
  
 public void setNov(int nov) {  
 this.nov = nov;  
 }  
}

Row View of each book

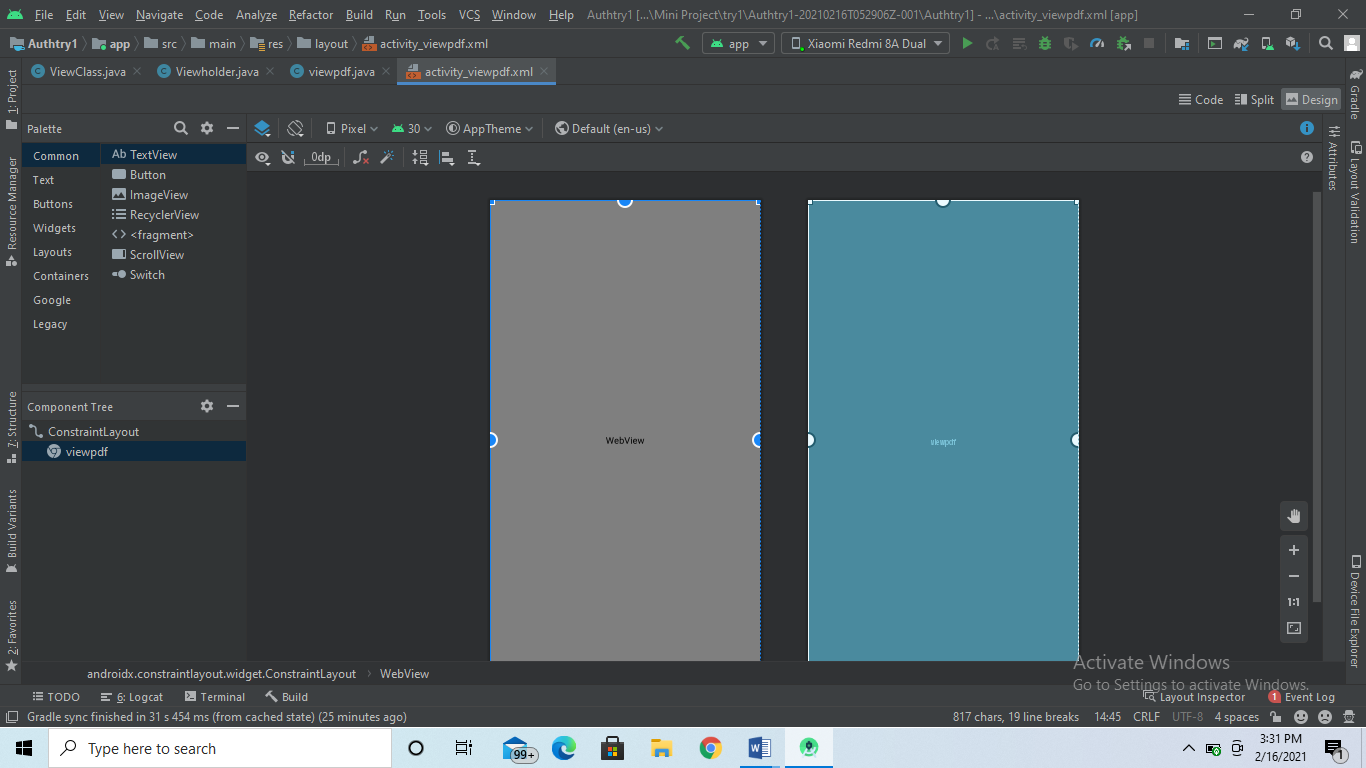
Java

package com.example.authtry1;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.os.Bundle;  
  
public class Row extends AppCompatActivity {  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_row*);  
 }  
}

View Pdf

Used To View pdf

XML

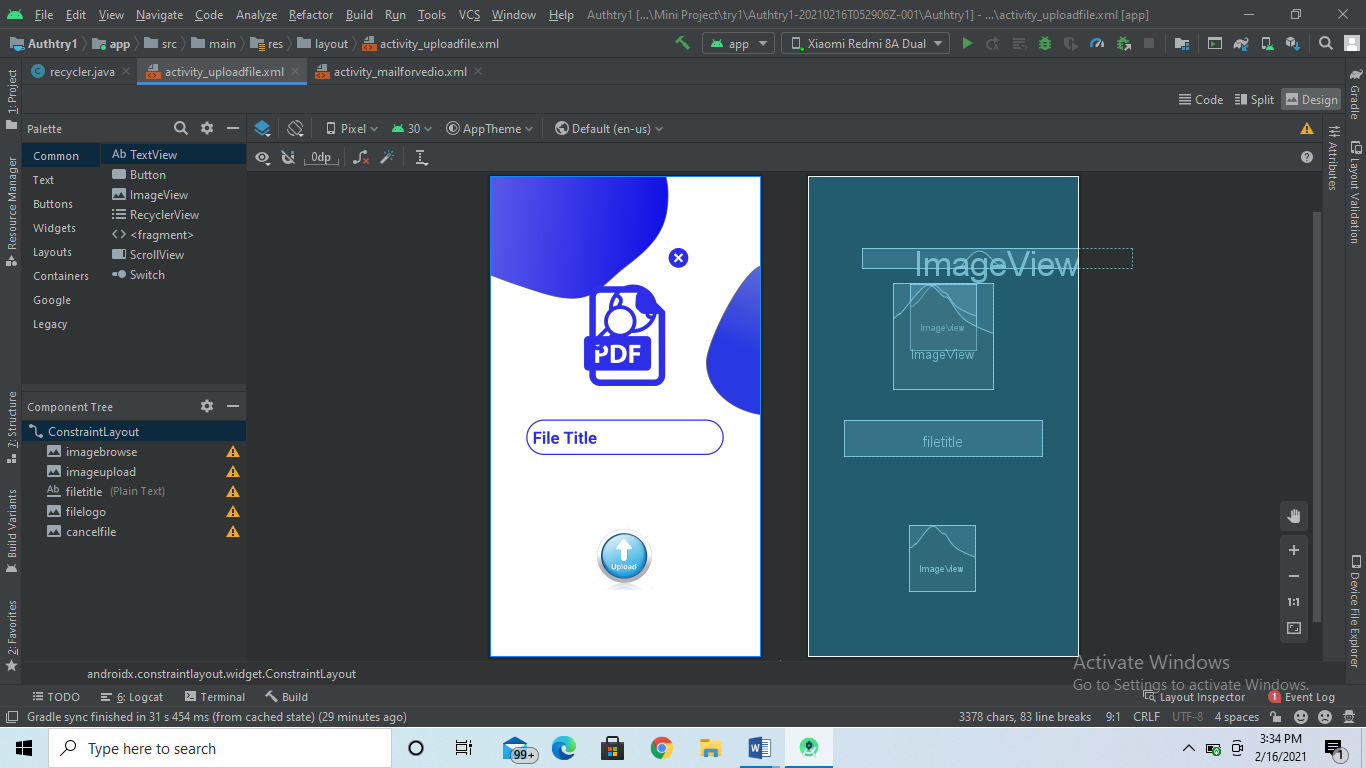
Java

package com.example.authtry1;  
  
import android.app.ProgressDialog;  
import android.graphics.Bitmap;  
import android.os.Bundle;  
import android.webkit.WebView;  
import android.webkit.WebViewClient;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import java.net.URLEncoder;  
  
public class viewpdf extends AppCompatActivity {  
  
 WebView pdfview;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_viewpdf*);  
  
  
 pdfview=(WebView)findViewById(R.id.*viewpdf*);  
 pdfview.getSettings().setJavaScriptEnabled(true);  
  
  
 String filename=getIntent().getStringExtra("filename");  
 String fileurl=getIntent().getStringExtra("fileurl");  
  
 final ProgressDialog pd=new ProgressDialog(this);  
 pd.setTitle(filename);  
 pd.setMessage("Opening....!!!");  
  
  
 pdfview.setWebViewClient(new WebViewClient()  
 {  
 @Override  
 public void onPageStarted(WebView view, String url, Bitmap favicon) {  
 super.onPageStarted(view, url, favicon);  
 pd.show();  
 }  
  
 @Override  
 public void onPageFinished(WebView view, String url) {  
 super.onPageFinished(view, url);  
 pd.dismiss();  
 }  
 });  
  
 String url="";  
 try {  
 url= URLEncoder.*encode*(fileurl,"UTF-8");  
 }catch (Exception ex)  
 {}  
  
 pdfview.loadUrl("http://docs.google.com/gview?embedded=true&url=" + url);  
  
 }  
  
  
}

Upload Pdf

Used To Upload pdf

XML

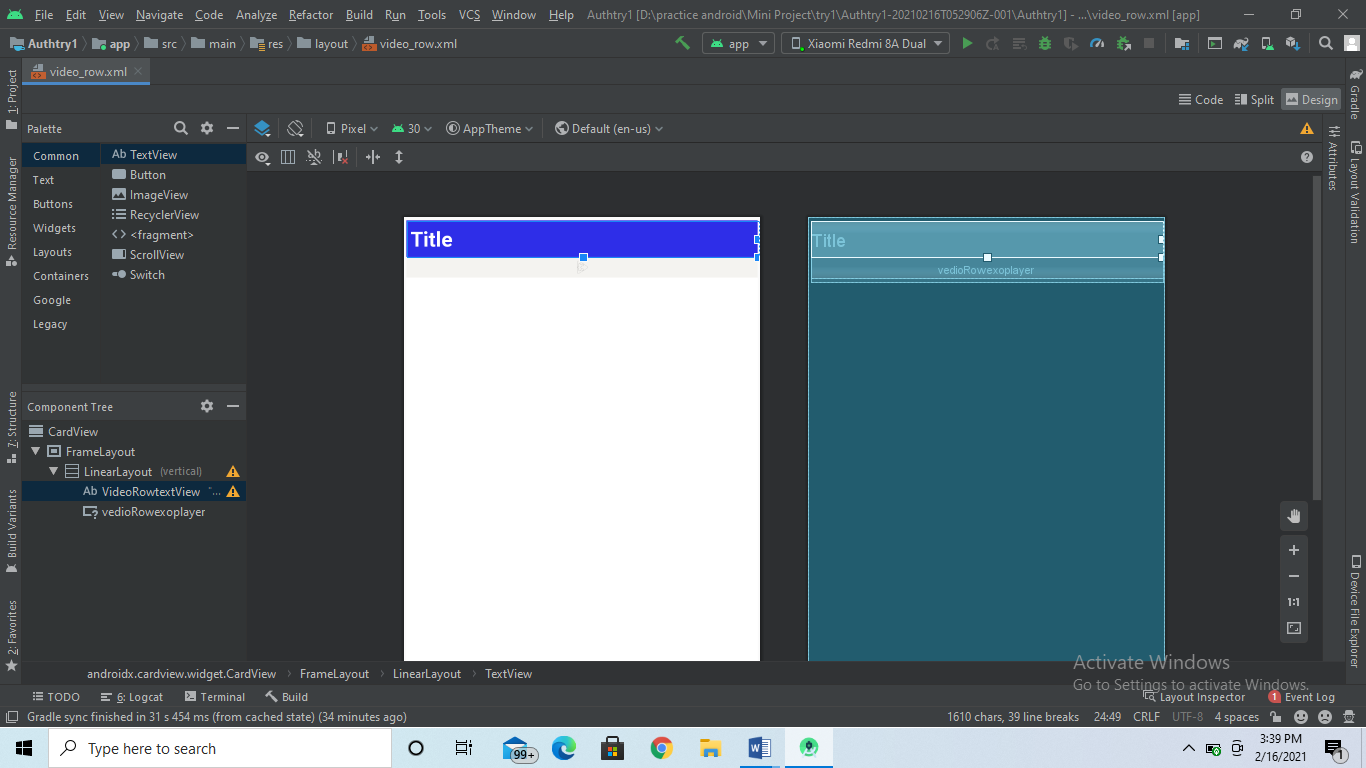


Java

package com.example.authtry1;  
  
import android.Manifest;  
import android.app.ProgressDialog;  
import android.content.Intent;  
import android.net.Uri;  
import android.os.Bundle;  
import android.view.View;  
import android.view.WindowManager;  
import android.widget.EditText;  
import android.widget.ImageView;  
import android.widget.Toast;  
  
import androidx.annotation.Nullable;  
import androidx.appcompat.app.AppCompatActivity;  
  
import com.google.android.gms.tasks.OnSuccessListener;  
import com.google.firebase.database.DatabaseReference;  
import com.google.firebase.database.FirebaseDatabase;  
import com.google.firebase.storage.FirebaseStorage;  
import com.google.firebase.storage.OnProgressListener;  
import com.google.firebase.storage.StorageReference;  
import com.google.firebase.storage.UploadTask;  
import com.karumi.dexter.Dexter;  
import com.karumi.dexter.PermissionToken;  
import com.karumi.dexter.listener.PermissionDeniedResponse;  
import com.karumi.dexter.listener.PermissionGrantedResponse;  
import com.karumi.dexter.listener.PermissionRequest;  
import com.karumi.dexter.listener.single.PermissionListener;  
  
public class uploadfile extends AppCompatActivity {  
 ImageView imagebrowse, imageupload, filelogo, cancelfile;  
 Uri filepath;  
  
 EditText filetitle;  
  
 StorageReference storageReference;  
 DatabaseReference databaseReference;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_uploadfile*);  
 getWindow().setFlags(WindowManager.LayoutParams.*FLAG\_FULLSCREEN*, WindowManager.LayoutParams.*FLAG\_FULLSCREEN*);  
  
 storageReference = FirebaseStorage.*getInstance*().getReference();  
 databaseReference = FirebaseDatabase.*getInstance*().getReference("mydocuments");  
  
 filetitle = findViewById(R.id.*filetitle*);  
  
 imagebrowse = findViewById(R.id.*imagebrowse*);  
 imageupload = findViewById(R.id.*imageupload*);  
  
 filelogo = findViewById(R.id.*filelogo*);  
 cancelfile = findViewById(R.id.*cancelfile*);  
  
  
 filelogo.setVisibility(View.*INVISIBLE*);  
 cancelfile.setVisibility(View.*INVISIBLE*);  
  
 cancelfile.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 filelogo.setVisibility(View.*INVISIBLE*);  
 cancelfile.setVisibility(View.*INVISIBLE*);  
 imagebrowse.setVisibility(View.*VISIBLE*);  
 }  
 });  
  
  
 imagebrowse.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 Dexter.*withContext*(getApplicationContext())  
 .withPermission(Manifest.permission.*READ\_EXTERNAL\_STORAGE*)  
 .withListener(new PermissionListener() {  
 @Override  
 public void onPermissionGranted(PermissionGrantedResponse permissionGrantedResponse) {  
 Intent intent = new Intent();  
 intent.setType("application/pdf");  
 intent.setAction(Intent.*ACTION\_GET\_CONTENT*);  
 startActivityForResult(Intent.*createChooser*(intent, "Select Pdf Files"), 101);  
 }  
  
 @Override  
 public void onPermissionDenied(PermissionDeniedResponse permissionDeniedResponse) {  
  
 }  
  
 @Override  
 public void onPermissionRationaleShouldBeShown(PermissionRequest permissionRequest, PermissionToken permissionToken) {  
 permissionToken.continuePermissionRequest();  
 }  
 }).check();  
 }  
 });  
  
 imageupload.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 processupload(filepath);  
 }  
 });  
 }  
  
  
 @Override  
 protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {  
 super.onActivityResult(requestCode, resultCode, data);  
  
 if (requestCode == 101 && resultCode == *RESULT\_OK*) {  
 filepath = data.getData();  
 filelogo.setVisibility(View.*VISIBLE*);  
 cancelfile.setVisibility(View.*VISIBLE*);  
 imagebrowse.setVisibility(View.*INVISIBLE*);  
 }  
 }  
  
  
 public void processupload(Uri filepath) {  
 final ProgressDialog pd = new ProgressDialog(this);  
 pd.setTitle("File Uploading....!!!");  
 pd.show();  
  
 final StorageReference reference = storageReference.child("uploads/" + System.*currentTimeMillis*() + ".pdf");  
 reference.putFile(filepath)  
 .addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {  
 @Override  
 public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {  
  
 reference.getDownloadUrl().addOnSuccessListener(new OnSuccessListener<Uri>() {  
 @Override  
 public void onSuccess(Uri uri) {  
  
 model obj = new model(filetitle.getText().toString(), uri.toString(), 0, 0, 0);  
 databaseReference.child(databaseReference.push().getKey()).setValue(obj);  
  
 pd.dismiss();  
 Toast.*makeText*(getApplicationContext(), "File Uploaded", Toast.*LENGTH\_LONG*).show();  
  
 filelogo.setVisibility(View.*INVISIBLE*);  
 cancelfile.setVisibility(View.*INVISIBLE*);  
 imagebrowse.setVisibility(View.*VISIBLE*);  
 filetitle.setText("");  
 }  
 });  
  
 }  
 })  
 .addOnProgressListener(new OnProgressListener<UploadTask.TaskSnapshot>() {  
 @Override  
 public void onProgress(UploadTask.TaskSnapshot taskSnapshot) {  
 float percent = (100 \* taskSnapshot.getBytesTransferred()) / taskSnapshot.getTotalByteCount();  
 pd.setMessage("Uploaded :" + (int) percent + "%");  
 }  
 });  
  
  
 }  
}

Video Class

Used to upload and view videos

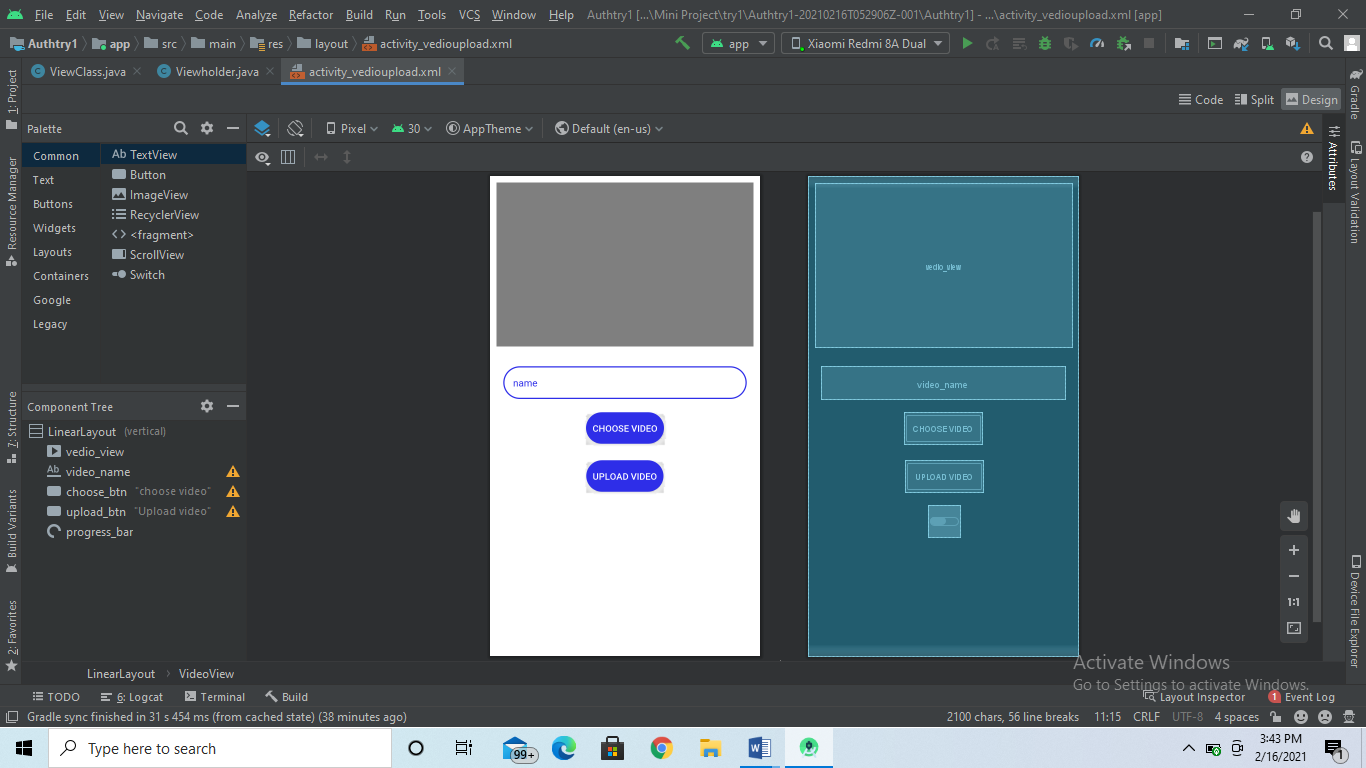
Java

package com.example.authtry1;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.annotation.NonNull;  
import androidx.annotation.Nullable;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import android.app.Application;  
import android.content.ContentResolver;  
import android.content.Intent;  
import android.net.Uri;  
import android.os.Bundle;  
import android.util.Log;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.webkit.MimeTypeMap;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.MediaController;  
import android.widget.ProgressBar;  
import android.widget.Toast;  
import android.widget.VideoView;  
  
import com.google.android.gms.tasks.OnFailureListener;  
import com.google.android.gms.tasks.OnSuccessListener;  
import com.google.firebase.database.DatabaseReference;  
import com.google.firebase.database.FirebaseDatabase;  
import com.google.firebase.ktx.Firebase;  
import com.google.firebase.storage.FirebaseStorage;  
import com.google.firebase.storage.StorageReference;  
import com.google.firebase.storage.StorageTask;  
import com.google.firebase.storage.UploadTask;  
  
import java.io.File;  
  
  
import android.os.Bundle;  
  
public class Vedioupload extends AppCompatActivity {  
 private static final int *PICK\_VIDEO\_REQUEST* = 1;  
  
  
 private Button choosebtn;  
 private Button uploadbtn;  
 private ProgressBar progressBar;  
 private VideoView videoView;  
 private EditText title;  
 private Uri url;  
 MediaController mediaController;  
 private StorageReference mStorageRef;  
 private DatabaseReference mDataBaseRef;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_vedioupload*);  
 choosebtn = findViewById(R.id.*choose\_btn*);  
 uploadbtn = findViewById(R.id.*upload\_btn*);  
 videoView = findViewById(R.id.*vedio\_view*);  
 progressBar = findViewById(R.id.*progress\_bar*);  
 title = findViewById(R.id.*video\_name*);  
  
 mediaController = new MediaController(this);  
  
 mStorageRef = FirebaseStorage.*getInstance*().getReference("video");  
 mDataBaseRef = FirebaseDatabase.*getInstance*().getReference("video");  
   
 videoView.setMediaController(mediaController);  
 mediaController.setAnchorView(videoView);  
 videoView.start();  
  
  
 choosebtn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 ChooseVideo();  
 }  
 });  
  
 uploadbtn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 UploadVideo();  
 }  
 });  
  
 }  
  
 private void ChooseVideo() {  
 Intent intent = new Intent();  
 intent.setType("video/\*");  
 intent.setAction(Intent.*ACTION\_GET\_CONTENT*);  
 startActivityForResult(intent, *PICK\_VIDEO\_REQUEST*);  
 }  
  
 @Override  
 protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {  
 super.onActivityResult(requestCode, resultCode, data);  
  
 if (requestCode == *PICK\_VIDEO\_REQUEST* && resultCode == *RESULT\_OK* && data != null && data.getData() != null) {  
  
 url = data.getData();  
  
 videoView.setVideoURI(url);  
  
  
 }  
 }  
  
 private String getFileExtension(Uri videoUri) {  
 ContentResolver cR = getContentResolver();  
 MimeTypeMap mime = MimeTypeMap.*getSingleton*();  
 return mime.getExtensionFromMimeType(cR.getType(videoUri));  
 }  
  
 private void UploadVideo() {  
  
 progressBar.setVisibility(View.*VISIBLE*);  
 if (url != null) {  
 StorageReference reference = mStorageRef.child(System.*currentTimeMillis*() +  
 "." + getFileExtension(url));  
  
 reference.putFile(url)  
 .addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {  
 @Override  
 public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {  
 progressBar.setVisibility(View.*INVISIBLE*);  
 Toast.*makeText*(getApplicationContext(), "Upload successful", Toast.*LENGTH\_SHORT*).show();  
 Member member = new Member(title.getText().toString().trim(),  
 taskSnapshot.getUploadSessionUri().toString());  
 String UploadId = mDataBaseRef.push().getKey();  
 mDataBaseRef.child(UploadId).setValue(member);  
  
 }  
 })  
 .addOnFailureListener(new OnFailureListener() {  
 @Override  
 public void onFailure(@NonNull Exception e) {  
  
 Toast.*makeText*(getApplicationContext(), e.getMessage(), Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
  
  
 } else {  
 Toast.*makeText*(getApplicationContext(), "No file selected", Toast.*LENGTH\_SHORT*).show();  
 }  
  
  
 }  
}

View Holder

package com.example.authtry1;  
  
import android.app.Application;  
import android.net.Uri;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.TextView;  
  
import androidx.annotation.NonNull;  
import androidx.recyclerview.widget.RecyclerView;  
  
import com.firebase.ui.database.FirebaseRecyclerAdapter;  
import com.firebase.ui.database.FirebaseRecyclerOptions;  
import com.google.android.exoplayer2.ExoPlayerFactory;  
import com.google.android.exoplayer2.SimpleExoPlayer;  
import com.google.android.exoplayer2.extractor.DefaultExtractorsFactory;  
import com.google.android.exoplayer2.extractor.ExtractorsFactory;  
import com.google.android.exoplayer2.source.ExtractorMediaSource;  
import com.google.android.exoplayer2.source.MediaSource;  
import com.google.android.exoplayer2.trackselection.AdaptiveTrackSelection;  
import com.google.android.exoplayer2.trackselection.DefaultTrackSelector;  
import com.google.android.exoplayer2.trackselection.TrackSelector;  
import com.google.android.exoplayer2.ui.PlayerView;  
import com.google.android.exoplayer2.upstream.BandwidthMeter;  
import com.google.android.exoplayer2.upstream.DefaultBandwidthMeter;  
import com.google.android.exoplayer2.upstream.DefaultHttpDataSourceFactory;  
import com.google.android.exoplayer2.util.Log;  
  
public class Viewholder extends RecyclerView.ViewHolder {  
  
  
 SimpleExoPlayer exoPlayer;  
 PlayerView playerView;  
  
  
 public Viewholder(@NonNull View itemView) {  
 super(itemView);  
 }  
  
 public void setDetails2(final Application ctx, String name, final String url) {  
 TextView mTitleTv2 = itemView.findViewById(R.id.*rTitleTv2*);  
 playerView = itemView.findViewById(R.id.*ep\_video\_view*);  
  
  
 mTitleTv2.setText(name);  
 try {  
 BandwidthMeter bandwidthMeter = new DefaultBandwidthMeter.Builder(ctx).build();  
 TrackSelector trackSelector = new DefaultTrackSelector(new AdaptiveTrackSelection.Factory(bandwidthMeter));  
 exoPlayer = (SimpleExoPlayer) ExoPlayerFactory.*newSimpleInstance*(ctx);  
 Uri video = Uri.*parse*(url);  
 DefaultHttpDataSourceFactory dataSourceFactory = new DefaultHttpDataSourceFactory("video");  
 ExtractorsFactory extractorsFactory = new DefaultExtractorsFactory();  
 MediaSource mediaSource = new ExtractorMediaSource(video, dataSourceFactory, extractorsFactory, null, null);  
 playerView.setPlayer(exoPlayer);  
 exoPlayer.prepare(mediaSource);  
 exoPlayer.setPlayWhenReady(false);  
  
  
 } catch (Exception e) {  
 Log.*e*("ViewHolder2", "exoplayer error" + e.toString());  
 }  
  
  
 }  
  
}

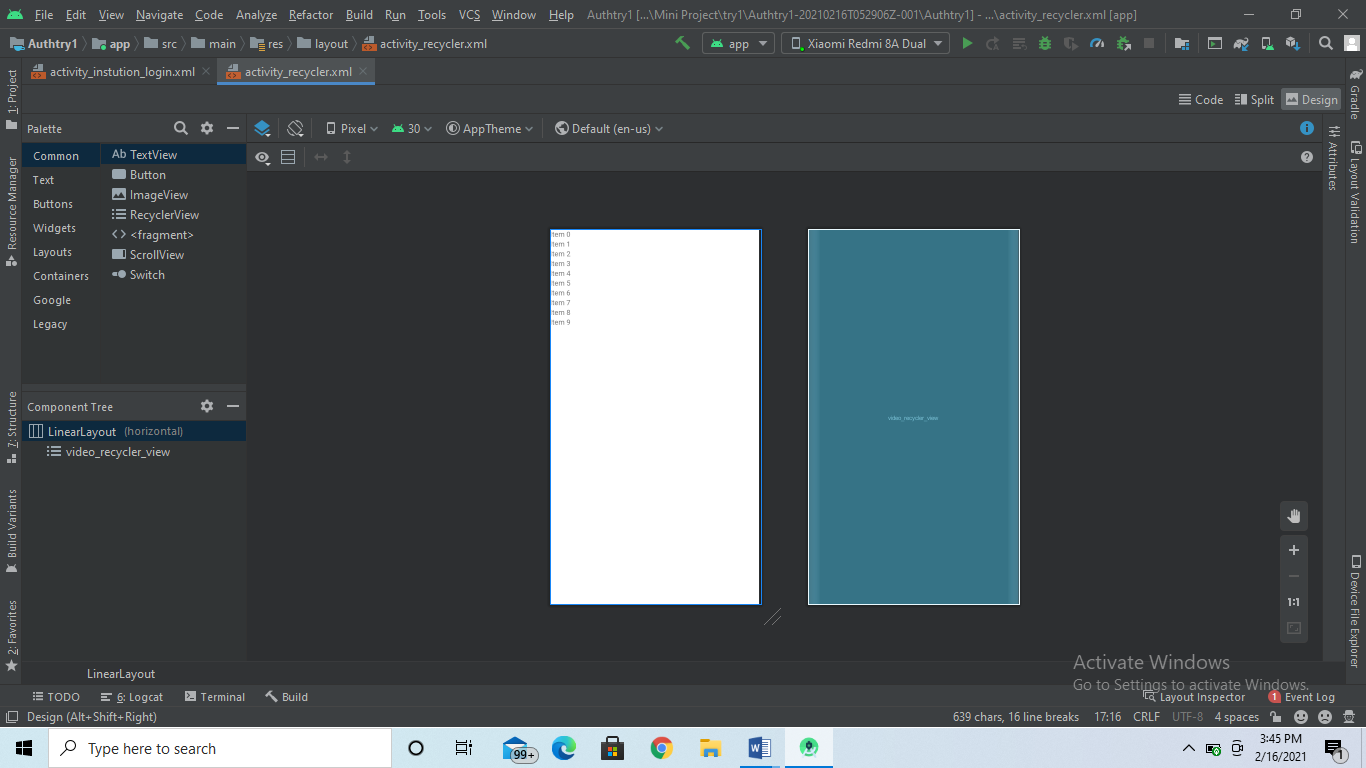
Video upload



Java

package com.example.authtry1;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.annotation.NonNull;  
import androidx.annotation.Nullable;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import android.app.Application;  
import android.content.ContentResolver;  
import android.content.Intent;  
import android.net.Uri;  
import android.os.Bundle;  
import android.util.Log;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.webkit.MimeTypeMap;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.MediaController;  
import android.widget.ProgressBar;  
import android.widget.Toast;  
import android.widget.VideoView;  
  
import com.google.android.gms.tasks.OnFailureListener;  
import com.google.android.gms.tasks.OnSuccessListener;  
import com.google.firebase.database.DatabaseReference;  
import com.google.firebase.database.FirebaseDatabase;  
import com.google.firebase.ktx.Firebase;  
import com.google.firebase.storage.FirebaseStorage;  
import com.google.firebase.storage.StorageReference;  
import com.google.firebase.storage.StorageTask;  
import com.google.firebase.storage.UploadTask;  
  
import java.io.File;  
  
  
import android.os.Bundle;  
  
public class Vedioupload extends AppCompatActivity {  
 private static final int *PICK\_VIDEO\_REQUEST* = 1;  
  
  
 private Button choosebtn;  
 private Button uploadbtn;  
 private ProgressBar progressBar;  
 private VideoView videoView;  
 private EditText title;  
 private Uri url;  
 MediaController mediaController;  
 private StorageReference mStorageRef;  
 private DatabaseReference mDataBaseRef;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_vedioupload*);  
 choosebtn = findViewById(R.id.*choose\_btn*);  
 uploadbtn = findViewById(R.id.*upload\_btn*);  
 videoView = findViewById(R.id.*vedio\_view*);  
 progressBar = findViewById(R.id.*progress\_bar*);  
 title = findViewById(R.id.*video\_name*);  
  
 mediaController = new MediaController(this);  
  
 mStorageRef = FirebaseStorage.*getInstance*().getReference("video");  
 mDataBaseRef = FirebaseDatabase.*getInstance*().getReference("video");  
 // mDataBaseRef= Firebase.database.getReference("Users")  
  
  
 videoView.setMediaController(mediaController);  
 mediaController.setAnchorView(videoView);  
 videoView.start();  
  
  
 choosebtn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 ChooseVideo();  
 }  
 });  
  
 uploadbtn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 UploadVideo();  
 }  
 });  
  
 }  
  
 private void ChooseVideo() {  
 Intent intent = new Intent();  
 intent.setType("video/\*");  
 intent.setAction(Intent.*ACTION\_GET\_CONTENT*);  
 startActivityForResult(intent, *PICK\_VIDEO\_REQUEST*);  
 }  
  
 @Override  
 protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {  
 super.onActivityResult(requestCode, resultCode, data);  
  
 if (requestCode == *PICK\_VIDEO\_REQUEST* && resultCode == *RESULT\_OK* && data != null && data.getData() != null) {  
  
 url = data.getData();  
  
 videoView.setVideoURI(url);  
  
  
 }  
 }  
  
 private String getFileExtension(Uri videoUri) {  
 ContentResolver cR = getContentResolver();  
 MimeTypeMap mime = MimeTypeMap.*getSingleton*();  
 return mime.getExtensionFromMimeType(cR.getType(videoUri));  
 }  
  
 private void UploadVideo() {  
  
 progressBar.setVisibility(View.*VISIBLE*);  
 if (url != null) {  
 StorageReference reference = mStorageRef.child(System.*currentTimeMillis*() +  
 "." + getFileExtension(url));  
  
 reference.putFile(url)  
 .addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {  
 @Override  
 public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {  
 progressBar.setVisibility(View.*INVISIBLE*);  
 Toast.*makeText*(getApplicationContext(), "Upload successful", Toast.*LENGTH\_SHORT*).show();  
 Member member = new Member(title.getText().toString().trim(),  
 taskSnapshot.getUploadSessionUri().toString());  
 String UploadId = mDataBaseRef.push().getKey();  
 mDataBaseRef.child(UploadId).setValue(member);  
  
 }  
 })  
 .addOnFailureListener(new OnFailureListener() {  
 @Override  
 public void onFailure(@NonNull Exception e) {  
  
 Toast.*makeText*(getApplicationContext(), e.getMessage(), Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
  
  
 } else {  
 Toast.*makeText*(getApplicationContext(), "No file selected", Toast.*LENGTH\_SHORT*).show();  
 }  
  
  
 }  
}

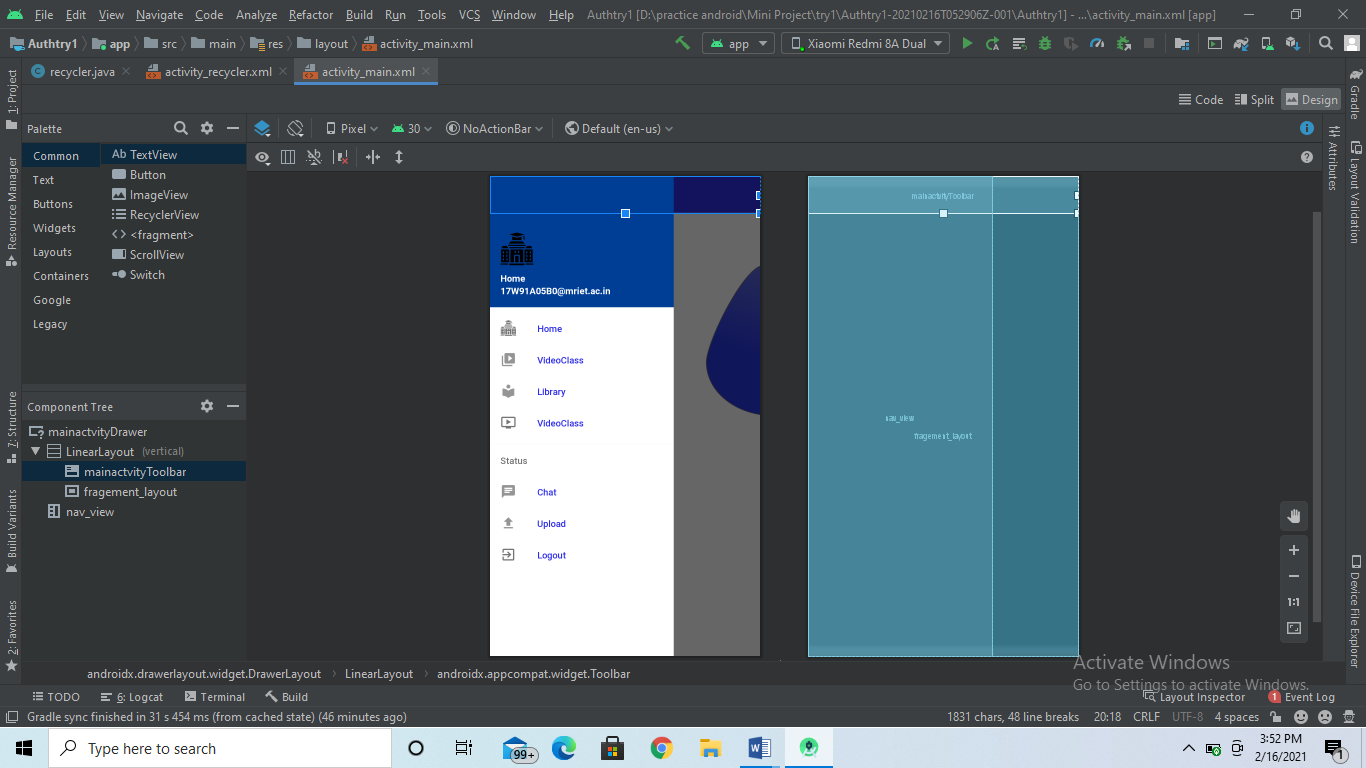
Recycler view

java

package com.example.authtry1;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import android.app.Application;  
import android.net.Uri;  
import android.os.Bundle;  
import android.provider.ContactsContract;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.TextView;  
  
import com.firebase.ui.database.FirebaseRecyclerAdapter;  
import com.firebase.ui.database.FirebaseRecyclerOptions;  
  
import com.google.firebase.database.DatabaseReference;  
import com.google.firebase.database.FirebaseDatabase;  
  
  
import android.os.Bundle;  
  
public class recycler extends AppCompatActivity {  
 public RecyclerView mRecyclerView2;  
 FirebaseDatabase mFirebaseDatabase2;  
 DatabaseReference mref2;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_recycler*);  
 mRecyclerView2 = findViewById(R.id.*video\_recycler\_view*);  
 mRecyclerView2.setHasFixedSize(true);  
 mRecyclerView2.setLayoutManager(new LinearLayoutManager(this));  
 mFirebaseDatabase2 = FirebaseDatabase.*getInstance*();  
 mref2 = mFirebaseDatabase2.getReference("video");  
 }  
  
 @Override  
 protected void onStart() {  
 super.onStart();  
  
 FirebaseRecyclerOptions<Model2> options =  
 new FirebaseRecyclerOptions.Builder<Model2>()  
 .setQuery(mref2, Model2.class)  
 .build();  
  
 FirebaseRecyclerAdapter<Model2, Viewholder> firebaseRecyclerAdapter =  
 new FirebaseRecyclerAdapter<Model2, Viewholder>(options) {  
 @Override  
 protected void onBindViewHolder(@NonNull Viewholder holder, int position, @NonNull Model2 model) {  
  
 holder.setDetails2(getApplication(), model.getName(), model.getUrl());  
 }  
  
 @NonNull  
 @Override  
 public Viewholder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
 View view = LayoutInflater.*from*(parent.getContext())  
 .inflate(R.layout.*activity\_row*, parent, false);  
  
  
 return new Viewholder(view);  
 }  
 };  
  
 firebaseRecyclerAdapter.startListening();  
 mRecyclerView2.setAdapter(firebaseRecyclerAdapter);  
  
  
 }  
  
  
}

Navigation drawer

Main Activity



Java

package com.example.authtry1;  
  
import android.content.DialogInterface;  
import android.content.Intent;  
import android.os.Bundle;   
import android.view.MenuItem;  
import android.view.View;  
import android.view.View.OnClickListener;  
import android.widget.Button;  
import android.widget.Toast;  
  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.ActionBarDrawerToggle;  
import androidx.appcompat.app.AlertDialog;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.appcompat.widget.Toolbar;  
import androidx.core.view.GravityCompat;  
import androidx.drawerlayout.widget.DrawerLayout;  
  
import com.google.android.material.navigation.NavigationView;  
import com.google.firebase.auth.FirebaseAuth;  
  
  
public class MainActivity extends AppCompatActivity implements NavigationView.OnNavigationItemSelectedListener {  
 private DrawerLayout drawerLayout;  
 AlertDialog.Builder builder;  
 FirebaseAuth auth;  
 Button button;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 Toolbar toolbar = (Toolbar) findViewById(R.id.*mainactvityToolbar*);  
 setSupportActionBar(toolbar);  
 auth = FirebaseAuth.*getInstance*();  
 builder = new AlertDialog.Builder(this);  
 drawerLayout = findViewById(R.id.*mainactvityDrawer*);  
 NavigationView navigationView = findViewById(R.id.*nav\_view*);  
 navigationView.setNavigationItemSelectedListener(this);  
 ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(this, drawerLayout, toolbar, R.string.*navigation\_drawer\_open*, R.string.*navigation\_drawer\_close*);  
 drawerLayout.addDrawerListener(toggle);  
 toggle.syncState();  
 if (savedInstanceState == null) {  
 getSupportFragmentManager().beginTransaction().replace(R.id.*fragement\_layout*,  
 new ClassFragment()).commit();  
 navigationView.setCheckedItem(R.id.*nav\_class*);  
 }  
 }  
  
  
 @Override  
 public boolean onNavigationItemSelected(@NonNull MenuItem item) {  
 Intent i;  
 switch (item.getItemId()) {  
 case R.id.*nav\_class*:  
 getSupportFragmentManager().beginTransaction().replace(R.id.*fragement\_layout*,  
 new ClassFragment()).commit();  
 break;  
 case R.id.*nav\_Insitutechat*:  
 Toast.*makeText*(this, "Urder Development", Toast.*LENGTH\_SHORT*).show();  
 getSupportFragmentManager().beginTransaction().replace(R.id.*fragement\_layout*,  
 new ChatFragment()).commit();  
 break;  
 case R.id.*nav\_Library*:  
 i = new Intent(MainActivity.this, BooksActivity.class);  
 startActivity(i);  
 break;  
 case R.id.*nav\_liveClass*:  
 i = new Intent(MainActivity.this, Vedioupload.class);  
 startActivity(i);  
 break;  
 case R.id.*videoClasses*:  
 i = new Intent(MainActivity.this, recycler.class);  
 startActivity(i);  
  
 break;  
 case R.id.*nav\_Syllabus*:  
 i = new Intent(MainActivity.this, uploadfile.class);  
 startActivity(i);  
 break;  
 case R.id.*Logout*:  
 auth.signOut();  
 i = new Intent(MainActivity.this, InstutionLoginActivity.class);  
 startActivity(i);  
 finish();  
 break;  
 default:  
 getSupportFragmentManager().beginTransaction().replace(R.id.*fragement\_layout*,  
 new ClassFragment()).commit();  
  
 }  
 drawerLayout.closeDrawer(GravityCompat.*START*);  
 return true;  
 }  
  
 @Override  
 public void onBackPressed() {  
 if (drawerLayout.isDrawerOpen(GravityCompat.*START*)) {  
 drawerLayout.closeDrawer(GravityCompat.*START*);  
 } else {  
 builder.setMessage("Welcome to AD").setTitle("AlertDialog");  
 builder.setMessage("Do you want to close this Application?")  
 .setCancelable(false)  
 .setPositiveButton("yes", new DialogInterface.OnClickListener() {  
 @Override  
 public void onClick(DialogInterface dialog, int which) {  
 finish();  
 }  
 })  
 .setNegativeButton("No", new DialogInterface.OnClickListener() {  
 @Override  
 public void onClick(DialogInterface dialog, int which) {  
 dialog.cancel();  
 }  
 });  
 AlertDialog alertDialog = builder.create();  
 alertDialog.setTitle("Alert Dialog ex");  
 alertDialog.show();  
 super.onBackPressed();  
 }  
 }  
}

**8. SYSTEM TEST**

**INTRODUCTION**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail unacceptably. There are various types of test. Each test type addresses a specific testing requirement.

**TYPES OF TESTS**

⊗**Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at the component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

⊗**Integration testing**

Integration tests are designed to test integrated software components to determine if they run as one program. Testing is event-driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

⊗**Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centred on the following items:

Valid Input  
Invalid Input Functions  
Output Systems/Procedures

: Identified classes of valid input must be accepted.  
: Identified classes of invalid input must be rejected.  
: Identified functions must be exercised.  
: Identified classes of application outputs must be exercised. : Interfacing systems or procedures must be invoked.

Organization and preparation of functional tests are focused on requirements, key functions, or special test cases. Also, systematic coverage about identifies Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

⊗**System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

⊗**White Box Testing**

White Box Testing is a testing in which the software tester knows the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black-box level.

⊗**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, like most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

### **TEST RESULTS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test case name** | **Testcase description** | **Expected value** | **Actual value** | **Result** |
|  | The user, as well as |  |  |  |
| Login Test | the admin, must be able to log in. A proper description of  the error must be displayed on failure. | The system must be easy to use and must be reliable. | The system was  reliable | Ready for deployment. |
|  |  |  |  |  |
|  | A new user must be |  |  |  |
| Sign In Test | able to create a new account and a new user must be taken to login page. | The system must function properly work useful prompts on error. | The system  was reliable | Ready for deployment. |
|  |  |  |  |  |
|  |  |  |  |  |
| Admin  Operation Test | To Check whether the admin can upload and change Values in Database | The admin must have an easy to use system to manage the data in Database. | The system was reliable | Ready for deployment. |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Database Test | To check whether the data is Properly Added and Retrieved from the Database  . | The user and Admin must have an easy to use Database to Upload and Retrieve the Data. | The system was reliable. | Ready for deployment. |
| Video  Activity Test | To check whether the functions linked with videos are properly. | The Activity must show current locations and locations based on Video | The System was Reliable | Ready for deployment |
| Application  Test | To check whether the activities working together Properly and giving proper results. | The module must calculate the quality score of a given image. | The system was  working properly. | Ready for deployment |

**CONCLUSION**

Testing is an especially important phase during the development of a project. It helps us find bugs and unwanted issues within the system. During this phase, we found some bugs in the system that we could easily fix. This helped us determine if the system was ready for real- world use. After rigorous testing, we could find that the system is ready for deployment.