**

**JK Lakshmipat University**

**DATABASE MANAGEMENT SYSTEM**

SUBMITTED BY – Indrajit Roy

Tanishka Nagal

Rupali Goyal

Priyanshi Mehta

Shrey Chechani

Section – B

Date of Submission – 17thNovember, 2024

Submitted To – Dr. S Taruna

Anamika Satrawala

**PROJECT REPORT   
OF   
FEEDBACK**

**MANAGEMENT   
SYSTEM**

**TABLE OF CONTENTS**

1.ABSTRACT ……………………………………………………… 4

2.FEATURES ………………………………………………………..5

3.INTRODUCTION …………………………………………………7

4.INTRODUCTION TO MYSQL ………………………………..…10

5. ER DIAGRAM……………………………………………….…...11

6. TABLES USED IN THE PROJECT ……………………….…….12

7. TECHNOLOGIES USED IN BACKEND ……………….………14

8. LOGIN PAGE AND ITS CODE  ………………………….……..17

9. DASHBOARD PAGE AND ITS CODE …………………..……..20

10. PROFILE FRAME AND ITS CODE …………………….…….22

11. FEEDBACK PAGE AND ITS CODE …………………..………24

12.CONCLUSIONS ………………………………………………..26

13.BIBLIOGRAPHY ……………………………………………….27

**Abstract**

The traditional focus on academic grades often overshadows the broader range of skills students develop outside the classroom. Recognizing this gap, *Beyond Grades* introduces an innovative feedback system designed to track and evaluate students' non-academic achievements. Through participation in extracurricular activities and receiving ratings and feedback from team leaders and organization heads, students gain valuable insights into their personal growth and key competencies. This report presents *Beyond Grades* as a holistic platform that highlights qualities essential for professional success, such as leadership, teamwork, resilience, and communication skills. This new approach supports a more complete evaluation of student potential, providing recruiters and placement cells with a comprehensive profile that extends beyond traditional academic metrics.

**FEATURES:**

1. **Activity Tracking**   
   Organize and document achievements across a range of extracurricular categories, including leadership, fitness, communication, and resilience. Each entry is structured to reflect not only the activity but the skills and contributions demonstrated in each context.
2. **Feedback-Based CGPA Calculation**Beyond Grades allows students to gather and review feedback provided by peers, team leaders, or organizing bodies. This feedback covers essential soft skills such as work ethic, responsibility, and teamwork, contributing to a cumulative CGPA that provides a more holistic assessment of each student's non-academic performance.
3. **Skills Overview**   
   Track and monitor critical attributes like emotional intelligence, multitasking, and interpersonal communication. This feature encourages continuous development in areas that contribute significantly to career readiness and personal growth.
4. **Insights for Development**   
   Leverage personalized feedback and CGPA results to identify areas of strength and potential growth. By regularly reviewing these insights, students can make informed decisions to pursue activities and roles that enhance specific skills.
5. **Organized Profile for Recruiters**   
   Beyond Grades presents an easy-to-navigate digital profile that clearly showcases a student’s skills, achievements, and personal growth beyond academics. This profile becomes an asset for recruiters, who can assess candidates with a balanced view of both intellectual and non-cognitive strengths.

**INTRODUCTION**

**Problem Statement**

Current student assessments are predominantly academic, overlooking essential skills like leadership, teamwork, resilience, and communication that contribute to personal and professional success. Traditional grading systems focus on intellectual achievements, limiting students’ ability to track soft skills and hindering recruiters from seeing a candidate’s full potential. This creates a gap, as students lack avenues to develop and showcase their broader competencies, while placement cells and recruiters miss valuable insights into these non-academic qualities.

**Beyond Grades** addresses this gap by creating a database-supported platform for tracking, managing, and evaluating extracurricular achievements. Students record participation in activities, and team leaders provide feedback on personal attributes, generating an extracurricular CGPA that reflects their holistic growth. This CGPA provides students with actionable insights while offering recruiters a structured view of non-academic skills, supporting more complete evaluations of candidates.

**Purpose and Importance**

As workplaces increasingly value well-rounded qualities like resilience, adaptability, and collaboration, the demand for holistic student profiles grows. Beyond Grades empowers students to take charge of their development, allowing them to track and manage their extracurricular activities in a structured format. By collecting feedback on teamwork, responsibility, and leadership, Beyond Grades offers a balanced measure of students’ strengths, promoting self-awareness and continuous improvement.

For recruiters and placement cells, Beyond Grades enables a more nuanced evaluation of candidates, combining academic achievements with critical soft skills. This approach aligns with the industry’s need for candidates who can work effectively in teams, communicate well, and lead when necessary.

**Project Overview**

Beyond Grades provides a flexible database system for organizing and assessing students’ extracurricular activities. Students participate in events, receive feedback on non-academic skills from team leaders, and log their achievements, all of which contribute to a holistic CGPA. This structured profile gives placement cells and recruiters a clear view of a candidate’s well-rounded strengths.

**Significance and Impact**

Beyond Grades redefines student evaluation by highlighting both academic knowledge and life skills such as leadership, empathy, and teamwork. This approach not only fosters personal growth and a continuous improvement mindset in students but also meets recruiters’ demand for candidates with balanced competencies. With Beyond Grades, placement teams gain insights into often-overlooked qualities, aiding in the selection of candidates who bring both technical and interpersonal strengths to the workplace.

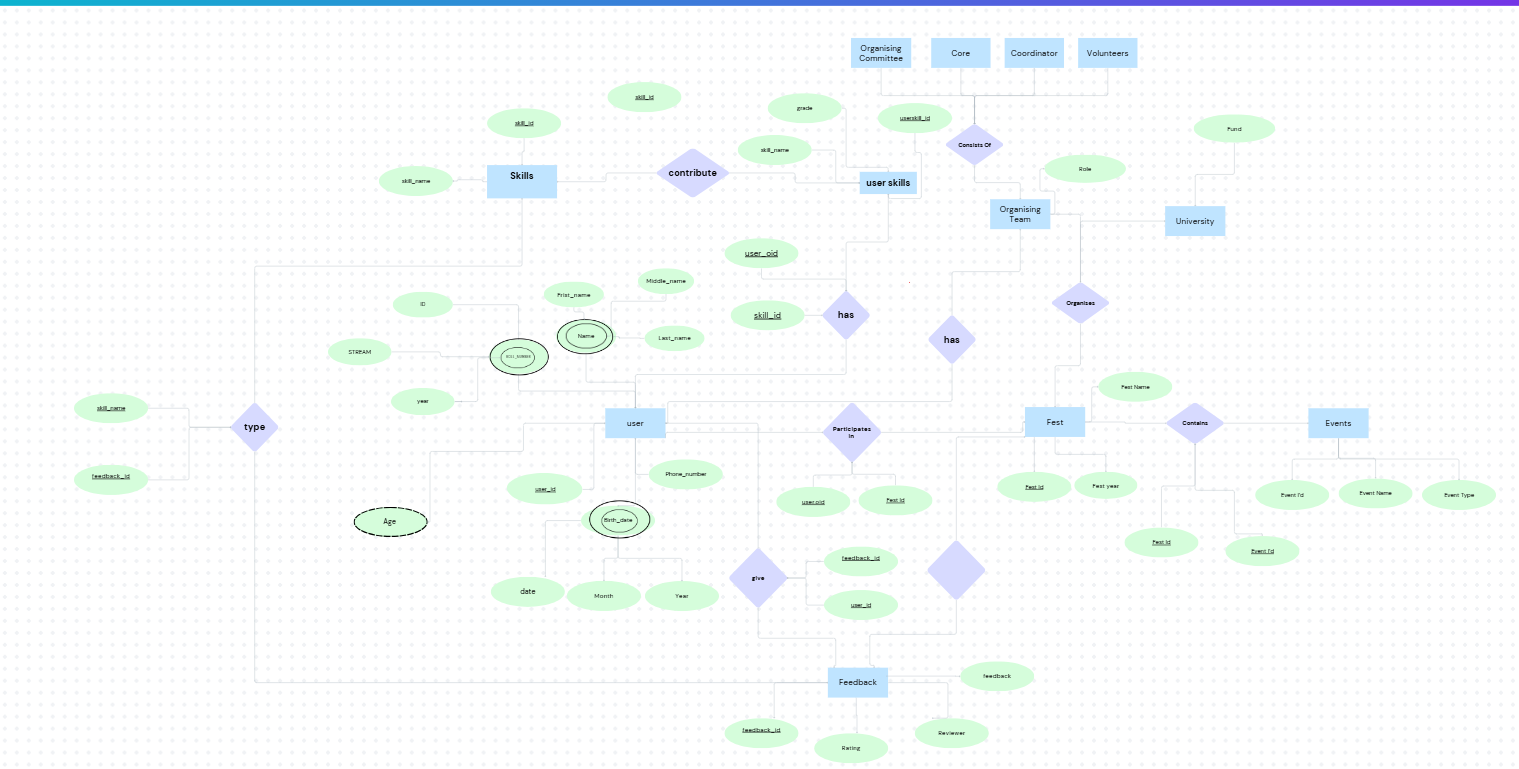


**Introduction to MySQL Database Management System**

**MySQL** is a popular, open-source relational database management system (RDBMS) originally developed by MySQL AB and now maintained by Oracle Corporation. Known for its ease of use, MySQL supports a wide range of applications and is available under an open-source license, making it customizable and cost-effective for various business needs.

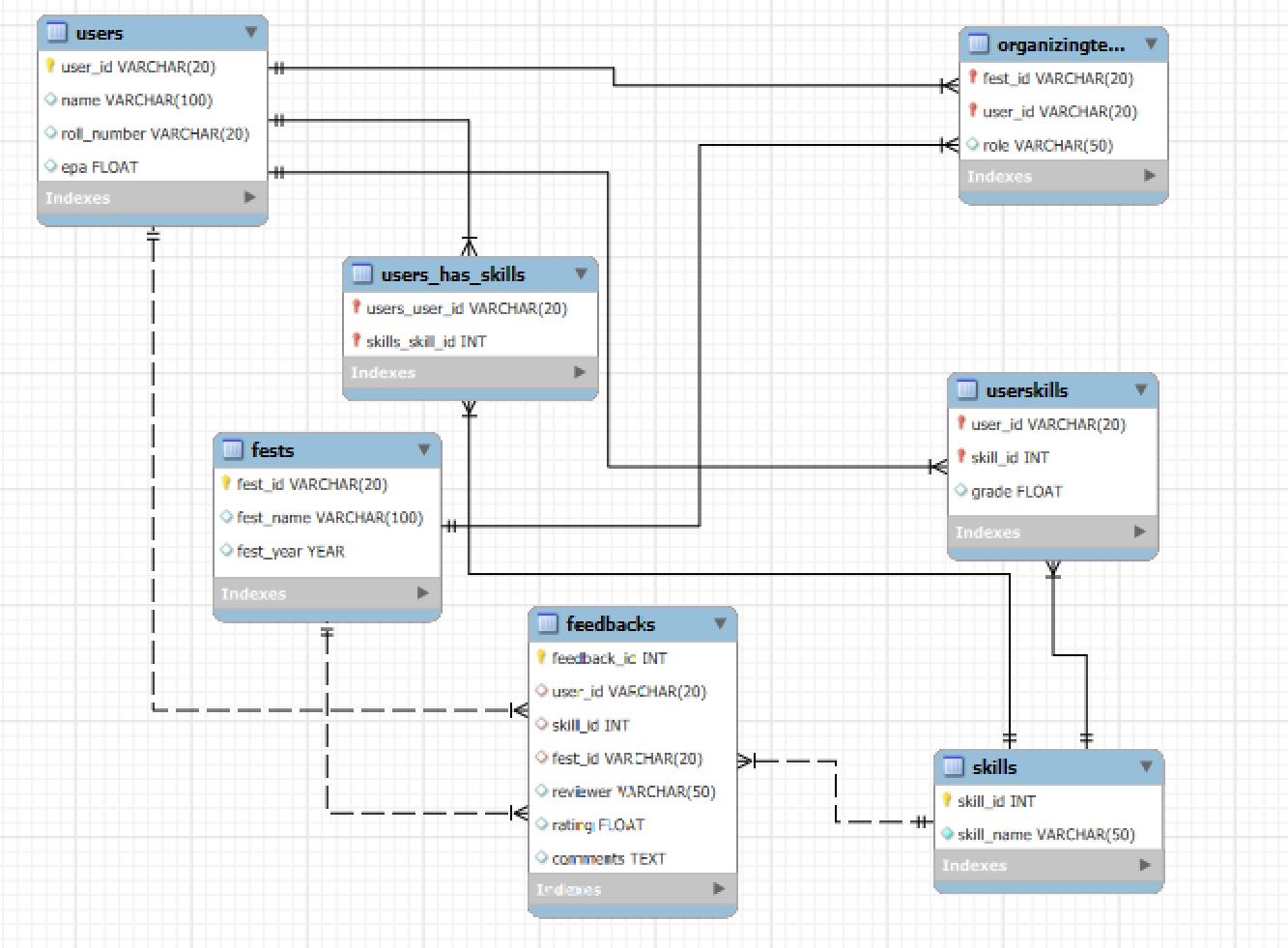
**Key Features of MySQL**

* Performance and Scalability: Optimized for fast query processing and scalable to handle large data sets.
* ANSI-SQL Compliance: Adheres to standard SQL, making it easy to learn and integrate with other systems.
* Cross-Platform Compatibility: Operates on multiple OS, including Windows, UNIX, and Linux, and supports languages like Java, PHP, and C++.
* Ease of Installation: User-friendly setup and configuration for rapid deployment in development or production.
* Large Database Support: Capable of managing large databases with partitioning, replication, and clustering.
* Security: Offers robust security with authentication, access control, and encryption.
* Community and Open-Source Flexibility: Extensive community support with customizable features for tailored solutions.

**DESIGN OF THE PROJECT**   
This project has been developed using MySQL software which is queries oriented.    
**ER-Diagram:**   


**Tables used in the project**

**Database Schema :**



***CREATE TABLE Users (***

***user\_id VARCHAR(20) PRIMARY KEY,***

***name VARCHAR(100),***

***roll\_number VARCHAR(20) UNIQUE,***

***epa FLOAT );***

***CREATE TABLE Skills (***

***skill\_id INT AUTO\_INCREMENT PRIMARY KEY,***

***skill\_name VARCHAR(50) NOT NULL );***

***CREATE TABLE Fests (***

***fest\_id VARCHAR(20) PRIMARY KEY,***

***fest\_name VARCHAR(100),***

***fest\_year YEAR );***

***CREATE TABLE UserSkills (***

***user\_id VARCHAR(20),***

***skill\_id INT,***

***grade INT,***

***PRIMARY KEY (user\_id, skill\_id),***

***FOREIGN KEY (user\_id) REFERENCES Users(user\_id),***

***FOREIGN KEY (skill\_id) REFERENCES Skills(skill\_id) );***

***CREATE TABLE OrganizingTeam (***

***fest\_id VARCHAR(20),***

***user\_id VARCHAR(20),***

***role VARCHAR(50),***

***PRIMARY KEY (fest\_id, user\_id),***

***FOREIGN KEY (fest\_id) REFERENCES Fests(fest\_id),***

***FOREIGN KEY (user\_id) REFERENCES Users(user\_id) );***

***CREATE TABLE Feedbacks (***

***feedback\_id INT AUTO\_INCREMENT PRIMARY KEY,***

***user\_id VARCHAR(20),***

***skill\_id INT,***

***fest\_id VARCHAR(20),***

***reviewer VARCHAR(50),***

***rating FLOAT,***

***comments TEXT,***

***FOREIGN KEY (user\_id) REFERENCES Users(user\_id),***

***FOREIGN KEY (skill\_id) REFERENCES Skills(skill\_id),***

***FOREIGN KEY (fest\_id) REFERENCES Fests(fest\_id) );***

**Technologies used in backend**

* **Firebase Realtime Database**

Firebase Realtime Database is a cloud-hosted NoSQL database that allows for seamless and instant synchronization of data across all connected clients. It helps developers store and retrieve data quickly, making it ideal for real-time applications like chat apps, live feeds, or multiplayer games. By eliminating the need for backend infrastructure setup, it simplifies app development and scalability.

* **Firebase Authentication**

Firebase Authentication simplifies the process of adding user authentication to apps, supporting various sign-in methods such as email/password, Google, Facebook, Twitter, and even custom providers. It handles the complexity of secure authentication, allowing developers to focus on app functionality. Firebase Authentication integrates smoothly with other Firebase services, ensuring a consistent and secure user experience.

* **Microsoft Authentication**

Microsoft Authentication allows users to sign in to your app using their Microsoft accounts. This integration is particularly useful in environments where Microsoft services (like Outlook, Office 365, or Teams) are commonly used. It enhances user experience by offering a widely recognized, secure sign-in method and is important for enterprise-level applications or apps targeting educational sectors.

* **Android Studio**

Android Studio is the official IDE for Android app development, offering a comprehensive suite of tools that streamline the development process. It includes a powerful code editor, graphical user interface (GUI) design tools, built-in emulators, and debugging utilities. Android Studio simplifies tasks like testing on multiple devices, optimizing app performance, and integrating with version control systems, making it the go-to platform for building Android apps.

* **Kotlin**

Kotlin is a modern programming language that runs on the Java Virtual Machine (JVM) and is officially supported for Android app development. Its clean, concise syntax reduces boilerplate code, making the codebase easier to maintain and read. Kotlin's enhanced safety features, such as null-safety, reduce the risk of runtime errors. It’s designed to be fully interoperable with Java, allowing developers to gradually migrate existing Java codebases to Kotlin.

* **Firebase BoM (Bill of Materials)**

The Firebase BoM (Bill of Materials) is a tool that helps manage Firebase dependencies in a consistent way. By using the BoM, developers ensure that they are using compatible versions of Firebase libraries without having to worry about version mismatches. This simplifies the setup and integration process, ensuring that different Firebase services (like Firestore, Authentication, etc.) work seamlessly together.

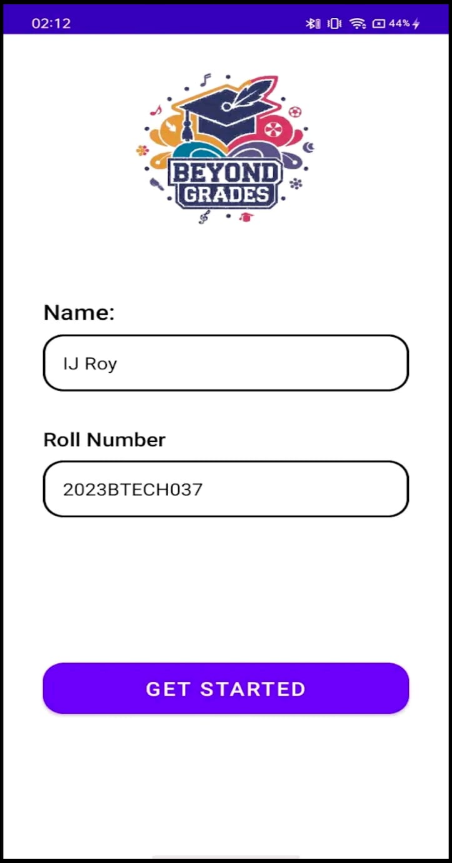
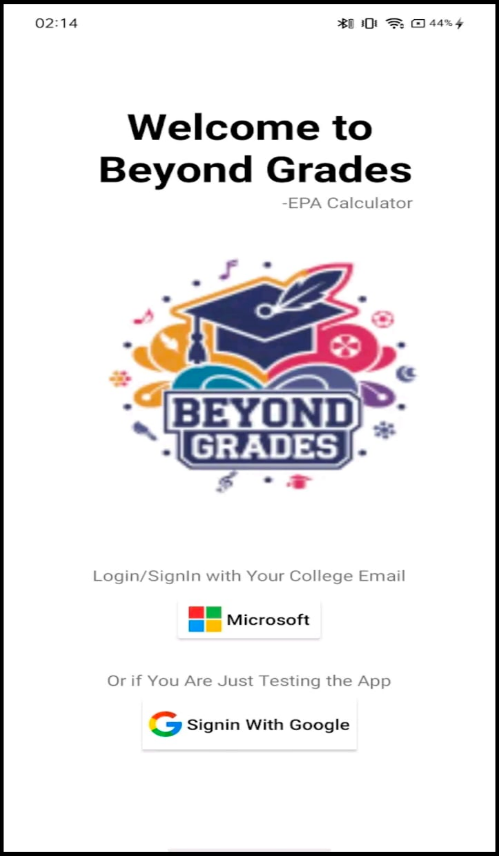
* **Bottom Navigation Menu**

The Bottom Navigation Menu is a common UI component that provides easy navigation between the main sections of an app, such as Home, Profile, and Settings. Positioned at the bottom of the screen, this navigation element helps users quickly switch between different areas of the app, improving the overall user experience by offering clear, one-tap access to essential features.

* **Fragments**

Fragments are modular UI components that represent portions of the user interface in an Android app. They allow for better organization of the app’s layout by breaking down complex UIs into smaller, reusable sections. Fragments can be dynamically added or replaced within an activity, offering greater flexibility in app design, especially for multi-screen or tablet layouts. Using fragments also promotes code reusability, as they can be shared across different activities or screens.

**Implementation of  
login page and its code**

Shape

***class SignIn : AppCompatActivity() {***

***private lateinit var auth: FirebaseAuth***

***private lateinit var database: FirebaseDatabase***

***private lateinit var binding: ActivitySignInBinding***

***private lateinit var googleSignInClient: GoogleSignInClient***

***private val RC\_SIGN\_IN = 1001 // Request code for Google Sign-In***

***override fun onCreate(savedInstanceState: Bundle?) {***

***super.onCreate(savedInstanceState)***

***enableEdgeToEdge()***

***binding = ActivitySignInBinding.inflate(layoutInflater)***

***setContentView(binding.root)***

***// Initialize FirebaseAuth and FirebaseDatabase***

***auth = FirebaseAuth.getInstance()***

***database = FirebaseDatabase.getInstance()***

***// Configure Google Sign-In***

***val gso = GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT\_SIGN\_IN)***

***.requestIdToken(getString(R.string.default\_web\_client\_id))***

***.requestEmail()***

***.build()***

***googleSignInClient = GoogleSignIn.getClient(this, gso)***

***binding.googleButton.setOnClickListener {***

***signInWithGoogle()***

***}***

***binding.microsoftButton.setOnClickListener {***

***if (auth.currentUser != null) {***

***checkUserData(auth.currentUser!!)***

***} else {***

***signInWithMicrosoft()***

***}***

***}***

***}***

**DashBoard And Its Code**

A screenshot of a phone

Description automatically generated

***class HomeFragment : Fragment() {***

***private var param1: String? = null***

***private var param2: String? = null***

***private lateinit var database: DatabaseReference***

***private lateinit var recyclerView: RecyclerView***

***private lateinit var adapter: SkillAdapter***

***private lateinit var overallEPA: TextView***

***override fun onCreate(savedInstanceState: Bundle?) {***

***super.onCreate(savedInstanceState)***

***arguments?.let {***

***param1 = it.getString(ARG\_PARAM1)***

***param2 = it.getString(ARG\_PARAM2)***

***}***

***database = FirebaseDatabase.getInstance().reference.child("users")***

***val userId = FirebaseAuth.getInstance().currentUser?.uid ?: return***

***database.child(userId).child("skills").get().addOnSuccessListener { snapshot ->***

***if (snapshot.exists()) {***

***var totalEPA = 0.0***

***var totalSkills = 0***

***for (skill in snapshot.children) {***

***val skillName = skill.key ?: continue***

***val feedbacks = skill.child("feedbacks")***

***var totalRating = 0.0***

***var count = 0***

***for (fest in feedbacks.children) {***

***for (feedback in fest.children) {***

***val rating = feedback.child("rating").getValue(Double::class.java) ?: 0.0***

***totalRating += rating***

***count++***

***}***

***}***

***val averageRating = if (count > 0) totalRating / count else 0.0***

***database.child(userId).child("skills").child(skillName).child("rating").setValue(averageRating)***

***totalEPA += averageRating***

***totalSkills++***

***}***

***if (totalSkills > 0) {***

***val overallEPAValue = totalEPA / totalSkills***

***database.child(userId).child("epa").setValue(overallEPAValue)***

***overallEPA.text = "%.2f".format(overallEPAValue)***

***}***

***loadSkillsIntoRecyclerView(userId)***

***}***

***}***

***}***

**Profile fragement and its Code**

A screenshot of a phone

Description automatically generated

***class ProfileFragment : Fragment() {***

***// Variables  
 private lateinit var database: DatabaseReference  
 private lateinit var eventsRecyclerView: RecyclerView  
 private lateinit var eventsAdapter: EventsAdapter  
 private val eventsList = mutableListOf<Event>()  
  
 // Add TextViews to show name and roll number  
 private lateinit var nameTextProfile: TextView  
 private lateinit var rollNumberTextProfile: TextView  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setHasOptionsMenu(true) // Enable options menu in this fragment  
 // Initialize Firebase Database reference  
 database = FirebaseDatabase.getInstance().reference  
 }***

***override fun onCreateView(  
 inflater: LayoutInflater, container: ViewGroup?,  
 savedInstanceState: Bundle?  
 ): View? {  
 // Inflate the layout for this fragment  
 val rootView = inflater.inflate(R.layout.fragment\_profile, container, false)  
  
 // Initialize RecyclerView and adapter  
 eventsRecyclerView = rootView.findViewById(R.id.recyclerViewEvents)  
 eventsAdapter = EventsAdapter(eventsList)  
 eventsRecyclerView.layoutManager = LinearLayoutManager(requireContext())  
 eventsRecyclerView.adapter = eventsAdapter  
  
 // Initialize TextViews for name and roll number  
 nameTextProfile = rootView.findViewById(R.id.nameTextProfile)  
 rollNumberTextProfile = rootView.findViewById(R.id.rollNumberTextProfile)  
  
 // Set up Toolbar as ActionBar  
 val toolbar: Toolbar = rootView.findViewById(R.id.toolbar)  
 (activity as AppCompatActivity).setSupportActionBar(toolbar)  
  
 // Fetch user data from Firebase  
 getUserData()  
  
 return rootView  
 }***

***override fun onCreateOptionsMenu(menu: Menu, menuInflater: MenuInflater) {  
 super.onCreateOptionsMenu(menu, menuInflater)  
 menuInflater.inflate(R.menu.menu\_profile\_options, menu)  
 }***

***override fun onOptionsItemSelected(item: MenuItem): Boolean {  
 return when (item.itemId) {  
 R.id.action\_edit -> {  
 openEditUserInfoDialog()  
 true  
 }  
 R.id.action\_sign\_out -> {  
 signOut()  
 true  
 }  
 else -> super.onOptionsItemSelected(item)  
 }  
 }***

**Feedback Fragements**

A screenshot of a chat

Description automatically generated  A screenshot of a phone

Description automatically generated

***class Feedback : AppCompatActivity() {  
  
 private lateinit var festNameinFeedback: TextView  
 private lateinit var festyearInFeedback: TextView  
 private lateinit var feedbackRecyclerView: RecyclerView  
 private lateinit var feedbackAdapter: FeedbackAdapter  
 private lateinit var feedbackList: MutableList<FeedbackDataClass>  
 private lateinit var database: FirebaseDatabase  
 private lateinit var usersRef: DatabaseReference  
 private lateinit var festsRef: DatabaseReference  
  
 private val roleLevels = mapOf(  
 "OC" to 1, // Top level  
 "Core" to 2, // Second level  
 "Coordinator" to 3, // Third level  
 "Volunteer" to 4 // Bottom level  
 )  
  
 private val skillComparisons = mapOf(  
 "Leadership Skills" to "all",  
 "Work Ethics" to "all",  
 "Resilience" to "all",  
 "Communication Skills" to "all",  
 "Critical Thinking Skills" to "all",  
 "Time Management Skills" to "all"  
 )  
  
 /\*  
 private val skillComparisons = mapOf(  
 "Leadership Skills" to "below",  
 "Work Ethics" to "same\_and\_above",  
 "Resilience" to "above\_and\_below",  
 "Communication Skills" to "same",  
 "Critical Thinking Skills" to "below",  
 "Time Management Skills" to "below\_and\_same"  
 )  
 \*/  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.activity\_feedback)  
  
 // Initialize views  
 festNameinFeedback = findViewById(R.id.festNameinFeedback)  
 festyearInFeedback = findViewById(R.id.festyearInFeedback)  
 feedbackRecyclerView = findViewById(R.id.feedbackRecyclerViewEvents)  
  
 // Initialize Firebase  
 database = FirebaseDatabase.getInstance()  
 festsRef = database.reference.child("fests")  
 usersRef = database.reference.child("users")  
  
 // Get festID from intent  
 val festID = intent.getStringExtra("festID").orEmpty()  
  
 if (festID.isEmpty()) {  
 showToast("Invalid fest ID")  
 finish()  
 return  
 }  
  
 fetchFestDetails(festID)  
  
 val sortSpinner: Spinner = findViewById(R.id.sortSpinner)  
  
// Array of sorting options  
 val sortingOptions = arrayOf("Sort by Skill Name", "Sort by User Name","Sort by User Role")  
  
 val adapter = ArrayAdapter(this, android.R.layout.simple\_spinner\_item, sortingOptions)  
 adapter.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item)  
 sortSpinner.adapter = adapter  
  
// Set up the listener to detect selection changes  
 sortSpinner.onItemSelectedListener = object : AdapterView.OnItemSelectedListener {  
 override fun onItemSelected(parentView: AdapterView<\*>, view: View?, position: Int, id: Long) {  
 when (position) {  
 0 -> feedbackAdapter.sortBySkill()  
 1 -> feedbackAdapter.sortByName()  
 2 -> feedbackAdapter.sortByPost()  
 }  
 }***

**CONCLUSIONS**

 The *Beyond Grades* project introduces an innovative approach to student evaluation by going beyond traditional academic grading to include vital personal and interpersonal skills developed through extracurricular activities. This database-driven feedback system offers a structured platform that enables students to track their achievements, gather feedback on essential qualities like leadership, teamwork, and resilience, and receive a holistic CGPA reflecting their non-academic growth. By incorporating features such as activity tracking, feedback-based assessments, and skill monitoring, *Beyond Grades* empowers students to build a well-rounded profile that supports their personal development and career readiness.

Furthermore, this platform provides placement cells and recruiters with an accessible view of candidates' non-academic competencies, helping them make more informed decisions. With the use of robust technologies such as MySQL for data management, Firebase for real-time data synchronization, and Android Studio for app development, *Beyond Grades* is an efficient and scalable solution for modern educational needs. This project ultimately fosters a more inclusive assessment model that values comprehensive growth and addresses the evolving demands of today's professional landscape.

**BIBLIOGRAPHY**

* **MySQL Documentation**  
  Oracle Corporation. *MySQL 8.0 Reference Manual*. Available at: <https://dev.mysql.com/doc/>
* **Firebase Documentation**  
  Google LLC. *Firebase Realtime Database and Authentication*. Available at: https://firebase.google.com/docs
* **Kotlin Language Reference**  
  JetBrains. *Kotlin Documentation*. Available at: https://kotlinlang.org/docs/reference/
* **Android Developer Guide**  
  Google LLC. *Android Studio and Fragments Documentation*. Available at: <https://developer.android.com>
* **Academic Journals on Holistic Education**
  + Slade, C. & Readman, K. (2013). "Addressing student diversity in higher education." *Journal of Educational Development*.
  + Brown, R. & Christensen, M. (2016). "Fostering non-cognitive skills through extracurricular activities." *Journal of Student Development*.

**THANK YOU**