ACHIEVEMENTS —> FRAGMENT

Achievements.class (java class)

```
package                          ;

public class Achievement {
   private String name;
   private String rollNumber;
   private String year;
   private String department;
   private String specialLab;
   private String eventName;
   private String projectTitle;
   private String eventDate;
   private String eventType;
   private String eventMode;
   private String achievementType;
   private String status;
   private String proofLink;

   // Default constructor (required for Firebase)
   public Achievement() {
   }

   // Parameterized Constructor
   public Achievement(String name, String rollNumber, String year, String
department, String specialLab,
                      String eventName, String projectTitle, String eventDate,
String eventType,
                      String eventMode, String achievementType, String status,
String proofLink) {
       this.name = name;
       this.rollNumber = rollNumber;
       this.year = year;
       this.department = department;
       this.specialLab = specialLab;
       this.eventName = eventName;
       this.projectTitle = projectTitle;
       this.eventDate = eventDate;
       this.eventType = eventType;
       this.eventMode = eventMode;
       this.achievementType = achievementType;
       this.status = status;
       this.proofLink = proofLink;

   }
```

```
    // Getters and Setters
    public String getName() { return name; }
    public void setName(String name) { this.name = name; }

    public String getRollNumber() { return rollNumber; }
    public void setRollNumber(String rollNumber) { this.rollNumber = rollNumber;
}

    public String getYear() { return year; }
    public void setYear(String year) { this.year = year; }

    public String getDepartment() { return department; }
    public void setDepartment(String department) { this.department = department;
}

    public String getSpecialLab() { return specialLab; }
    public void setSpecialLab(String specialLab) { this.specialLab = specialLab;
}

    public String getEventName() { return eventName; }
    public void setEventName(String eventName) { this.eventName = eventName; }

    public String getProjectTitle() { return projectTitle; }
    public void setProjectTitle(String projectTitle) { this.projectTitle =
projectTitle; }

    public String getEventDate() { return eventDate; }
    public void setEventDate(String eventDate) { this.eventDate = eventDate; }

    public String getEventType() { return eventType; }
    public void setEventType(String eventType) { this.eventType = eventType; }

    public String getEventMode() { return eventMode; }
    public void setEventMode(String eventMode) { this.eventMode = eventMode; }

    public String getAchievementType() { return achievementType; }
    public void setAchievementType(String achievementType) {
this.achievementType = achievementType; }

    public String getStatus() { return status; }
    public void setStatus(String status) { this.status = status; }

    public String getProofLink() { return proofLink; }
    public void setProofLink(String proofLink) { this.proofLink = proofLink; }
}
```

ACHIEVEMENTS.JAVA

```
package com.example.spllabportal;

import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.TextView;
import android.widget.Toast;


import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;


import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.android.play.integrity.internal.ac;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.Query;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.ValueEventListener;


import java.util.ArrayList;
import java.util.Calendar;
import java.util.Map;


public class Achievements extends Fragment {


   private FirebaseAuth mAuth;
   private FirebaseUser currentUser;
   private FloatingActionButton fabAddProject;
   private DatabaseReference databaseReference;
```

```
   private EditText etName, etRollNumber, etYear, etDepartment, etEventName,
etProjectTitle, etProof;
   private Spinner spinnerEventType, spinnerEventMode, spinnerAchievements;
   private RecyclerView recyclerViewAchievements;


   private TextView etSpecialLab, tvSelectedDate;
   private AchievementsAdapter achievementsAdapter;
   private ArrayList<Achievement> achievementsList;


   private String specialLab, RollNumber;

   public Achievements() {
       // Required empty public constructor
   }

   @Override
   public View onCreateView(LayoutInflater inflater, ViewGroup container,
Bundle savedInstanceState) {
       View view = inflater.inflate(R.layout.fragment_achievements, container,
false);
       View rootView = inflater.inflate(R.layout.activity_achievementsadd,
container, false);


       // Initialize Firebase Auth and Database
       mAuth = FirebaseAuth.getInstance();
       currentUser = mAuth.getCurrentUser();
       databaseReference = FirebaseDatabase.getInstance().getReference();
       fabAddProject = view.findViewById(R.id.fabAddProject);

       fabAddProject.setOnClickListener(new View.OnClickListener() {
           @Override
           public void onClick(View view) {
               // Create an intent to navigate to another page
               Intent intent = new Intent(getActivity(),
Achievementsadd.class);  // Use getActivity() instead of Achievements.this
               startActivity(intent);
            }
       });

       // Find Views
       etName = rootView.findViewById(R.id.etName);
       etRollNumber = rootView.findViewById(R.id.etRollnumber);
       etYear = rootView.findViewById(R.id.etyear);
       etDepartment = rootView.findViewById(R.id.etdepartment);
```

```
        etSpecialLab = rootView.findViewById(R.id.etSpecialLab);
        etProjectTitle = rootView.findViewById(R.id.et_project_title);
        etProof = rootView.findViewById(R.id.et_proof);
        tvSelectedDate = rootView.findViewById(R.id.tv_selected_date);
        etEventName = rootView.findViewById(R.id.et_event_name);  // This should
be an EditText
        spinnerEventType = rootView.findViewById(R.id.spinner_event_type);
        spinnerEventMode = rootView.findViewById(R.id.spinner_event_mode);
        spinnerAchievements = rootView.findViewById(R.id.spinner_achievements);


        recyclerViewAchievements =
view.findViewById(R.id.recyclerView_achievements);


        // Set up RecyclerView
        recyclerViewAchievements.setLayoutManager(new
LinearLayoutManager(getContext()));
        achievementsList = new ArrayList<>();
        achievementsAdapter = new AchievementsAdapter(achievementsList);
        recyclerViewAchievements.setAdapter(achievementsAdapter);


        // Set up Spinners
        ArrayAdapter<CharSequence> eventTypeAdapter =
ArrayAdapter.createFromResource(getContext(),
                R.array.Event_Type, android.R.layout.simple_spinner_item);

eventTypeAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdo
wn_item);
        spinnerEventType.setAdapter(eventTypeAdapter);


        ArrayAdapter<CharSequence> eventModeAdapter =
ArrayAdapter.createFromResource(getContext(),
                R.array.Event_Mode, android.R.layout.simple_spinner_item);

eventModeAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdo
wn_item);
        spinnerEventMode.setAdapter(eventModeAdapter);


        ArrayAdapter<CharSequence> achievementsAdapter =
ArrayAdapter.createFromResource(getContext(),
                R.array.Achievements, android.R.layout.simple_spinner_item);

achievementsAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dro
pdown_item);
        spinnerAchievements.setAdapter(achievementsAdapter);
```

```
        String teamLeaderName = currentUser.getDisplayName();


        DatabaseReference databaseReference =
FirebaseDatabase.getInstance().getReference();
        DatabaseReference studentsRef =
databaseReference.child("SpecialLab").child("Students");

        studentsRef.addListenerForSingleValueEvent(new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {
                if (dataSnapshot.exists()) {
                    boolean isFound = false; // To check if the user is found
                    for (DataSnapshot childSnapshot :
dataSnapshot.getChildren()) {
                        String name =
childSnapshot.child("Name").getValue(String.class);
                        if (name != null && name.equals(teamLeaderName)) {
                            // Fetch the details for the matched user
                            specialLab = childSnapshot.child("Special
Lab").getValue(String.class);
                            RollNumber =
childSnapshot.child("ID").getValue(String.class);
                            String Year =
childSnapshot.child("Year").getValue(String.class);
                            String Department =
childSnapshot.child("Department").getValue(String.class);

                            // Set values to TextViews if needed
                            etSpecialLab.setText(specialLab);
                            etRollNumber.setText(RollNumber);
                            etYear.setText(Year);
                            etDepartment.setText(Department);

                            isFound = true; // User found

                            // Fetch achievements ONLY after retrieving user
details
                            fetchSubmittedAchievements();
                            break; // Exit the loop once found
                        }
                    }
                    if (!isFound) {
                        Toast.makeText(getContext(), "No matching user found.",
Toast.LENGTH_SHORT).show();
                    }
                } else {
```

```
                    Toast.makeText(getContext(), "No students found.",
Toast.LENGTH_SHORT).show();
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {
                Toast.makeText(getContext(), "Error fetching data.",
Toast.LENGTH_SHORT).show();
            }
        });
        return view;
    }


    private void fetchSubmittedAchievements() {
        if (specialLab == null || specialLab.isEmpty()) {
            Toast.makeText(getContext(), "Special Lab is not available.",
Toast.LENGTH_SHORT).show();
            return;
        }

        if (RollNumber == null || RollNumber.isEmpty()) {
            Toast.makeText(getContext(), "Roll Number is not available.",
Toast.LENGTH_SHORT).show();
            return;
        }

        // Debugging log
        Log.d("AchievementsDebug", "Fetching for SpecialLab: " + specialLab + ",
RollNumber: " + RollNumber);

        DatabaseReference achievementsRef =
databaseReference.child("Achievements").child(specialLab).child(RollNumber);

        achievementsRef.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                achievementsList.clear();  // Clear to prevent duplicates

                // Debugging log
                Log.d("AchievementsDebug", "DataSnapshot: " +
dataSnapshot.toString());

                if (!dataSnapshot.exists()) {
                    Toast.makeText(getContext(), "No achievements found!",
Toast.LENGTH_SHORT).show();
                    return;
                }
```

```
                Achievement achievement =
dataSnapshot.getValue(Achievement.class);
                if (achievement != null) {
                    achievementsList.add(achievement);
                } else {
                    Log.e("AchievementDebug", "Achievement is null for snapshot:
" + dataSnapshot.toString());
                }


                achievementsAdapter.notifyDataSetChanged();

recyclerViewAchievements.scrollToPosition(achievementsList.size() - 1);
            }

            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {
                Toast.makeText(getContext(), "Error fetching achievements.",
Toast.LENGTH_SHORT).show();
            }
        });
    }


}
```

ADAPTER:

```
package                          ;

import android.util.Log;
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 java.util.List;

public class AchievementsAdapter extends
RecyclerView.Adapter<AchievementsAdapter.AchievementViewHolder> {

   private List<Achievement> achievementsList;

   public AchievementsAdapter(List<Achievement> achievementsList) {
       this.achievementsList = achievementsList;
   }

   @NonNull
   @Override
   public AchievementViewHolder onCreateViewHolder(@NonNull ViewGroup parent,
int viewType) {
       View itemView =
LayoutInflater.from(parent.getContext()).inflate(R.layout.item_achievement,
parent, false);
       return new AchievementViewHolder(itemView);
   }

   @Override
   public void onBindViewHolder(@NonNull AchievementViewHolder holder, int
position) {
       Achievement achievement = achievementsList.get(position);

       // Ensure achievement is not null
       if (achievement == null) {
           Log.e("AchievementsAdapter", "Achievement object is null at
position: " + position);
           return;
       }

       // Log data for debugging
       Log.d("AchievementsAdapter", "Binding: " + achievement.getEventName() +
" | Status: " + achievement.getStatus());
```

```
        // Set event name & status safely
        holder.tvName.setText(achievement.getName() != null ?
achievement.getName() : "No Name");
        holder.tvRollNumber.setText(achievement.getRollNumber() != null ?
achievement.getRollNumber() : "No Roll Number");
        holder.tvEventName.setText(achievement.getEventName() != null ?
achievement.getEventName() : "No Event Name");
        holder.tvYear.setText(achievement.getYear() != null ?
achievement.getYear() : "No Year");
        holder.tvDepartment.setText(achievement.getDepartment() != null ?
achievement.getDepartment() : "No Department");
        holder.tvSpecialLab.setText(achievement.getSpecialLab() != null ?
achievement.getSpecialLab() : "No Special Lab");
        holder.tvProjectTitle.setText(achievement.getProjectTitle() != null ?
achievement.getProjectTitle() : "No Project Title");
        holder.tvEventType.setText(achievement.getEventType() != null ?
achievement.getEventType() : "No Event Type");
        holder.tvEventMode.setText(achievement.getEventMode() != null ?
achievement.getEventMode() : "No Event Mode");
        holder.tvAchievementType.setText(achievement.getAchievementType() !=
null ? achievement.getAchievementType() : "No Achievement Type");
        holder.tvProofLink.setText(achievement.getProofLink() != null ?
achievement.getProofLink() : "No Proof Link");
        holder.tvStatus.setText(achievement.getStatus() != null ?
achievement.getStatus() : "No Status");

        // Set status color
        if ("Pending".equals(achievement.getStatus())) {

holder.tvStatus.setTextColor(holder.itemView.getContext().getResources().getCol
or(R.color.pending));
        } else if ("Approved".equals(achievement.getStatus())) {

holder.tvStatus.setTextColor(holder.itemView.getContext().getResources().getCol
or(R.color.approved));
        } else if ("Rejected".equals(achievement.getStatus())) {

holder.tvStatus.setTextColor(holder.itemView.getContext().getResources().getCol
or(R.color.rejected));
        } else {

holder.tvStatus.setTextColor(holder.itemView.getContext().getResources().getCol
or(R.color.teal_700)); // Default color
        }
    }

    @Override
    public int getItemCount() {
```

```
        return achievementsList != null ? achievementsList.size() : 0;
    }

    public void updateData(List<Achievement> newAchievementsList) {
        this.achievementsList = newAchievementsList;
        notifyDataSetChanged();
    }

    public class AchievementViewHolder extends RecyclerView.ViewHolder {

        public TextView tvName, tvRollNumber, tvEventName, tvStatus, tvYear,
tvDepartment,
                tvSpecialLab, tvProjectTitle, tvEventType, tvEventMode,
                tvAchievementType, tvProofLink;

        public AchievementViewHolder(@NonNull View itemView) {
            super(itemView);

            tvName = itemView.findViewById(R.id.tv_name);
            tvRollNumber = itemView.findViewById(R.id.tv_rollnumber);
            tvEventName = itemView.findViewById(R.id.tv_event_name);
            tvStatus = itemView.findViewById(R.id.tv_status);
            tvYear = itemView.findViewById(R.id.tv_year);
            tvDepartment = itemView.findViewById(R.id.tv_department);
            tvSpecialLab = itemView.findViewById(R.id.tv_specialLab);
            tvProjectTitle = itemView.findViewById(R.id.tv_projecttitle);
            tvEventType = itemView.findViewById(R.id.tv_eventType);
            tvEventMode = itemView.findViewById(R.id.tv_eventMode);
            tvAchievementType = itemView.findViewById(R.id.tv_achievementType);
            tvProofLink = itemView.findViewById(R.id.tv_proofLink);
        }
    }

}
```

NEW ADD:

```
package                          ;

import android.app.DatePickerDialog;
import android.app.Dialog;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.Spinner;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.DialogFragment;
import androidx.recyclerview.widget.RecyclerView;


import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.Query;
import com.google.firebase.database.ValueEventListener;

import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.Locale;

public class Achievementsadd extends AppCompatActivity {


   private FirebaseAuth mAuth;
   private FirebaseUser currentUser;

   private SimpleDateFormat sdf = new SimpleDateFormat("dd-MM-yyyy",
Locale.getDefault()); // FIXED
```

```
    private DatabaseReference databaseReference;
    private ImageView ivCalendarIcon;

    private EditText etName, etRollNumber, etYear, etDepartment, etEventName,
etProjectTitle, etProof;
    private Spinner spinnerEventType, spinnerEventMode, spinnerAchievements;

    private TextView etSpecialLab,tvSelectedDate;
    private Button btnSubmit;

    private String specialLab, name, rollNumber, year, department;

    public Achievementsadd() {
        // Required empty public constructor
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_achievementsadd);

        mAuth = FirebaseAuth.getInstance();
        currentUser = mAuth.getCurrentUser();
        databaseReference = FirebaseDatabase.getInstance().getReference();
        ivCalendarIcon = findViewById(R.id.ivCalendarIcon);
        // Find Views
        etName = findViewById(R.id.etName);
        etRollNumber =findViewById(R.id.etRollnumber);
        etYear = findViewById(R.id.etyear);
        etDepartment =findViewById(R.id.etdepartment);
        etSpecialLab = findViewById(R.id.etSpecialLab);
        etProjectTitle = findViewById(R.id.et_project_title);
        etProof =findViewById(R.id.et_proof);
        tvSelectedDate = findViewById(R.id.tv_selected_date);
        etEventName = findViewById(R.id.et_event_name);  // This should be an
EditText
        spinnerEventType = findViewById(R.id.spinner_event_type);
        spinnerEventMode = findViewById(R.id.spinner_event_mode);
        spinnerAchievements =findViewById(R.id.spinner_achievements);
        btnSubmit = findViewById(R.id.btn_submit);
        // Set up Spinners
        ArrayAdapter<CharSequence> eventTypeAdapter =
ArrayAdapter.createFromResource(this,
                R.array.Event_Type, android.R.layout.simple_spinner_item);
        spinnerEventType.setAdapter(eventTypeAdapter);

        ArrayAdapter<CharSequence> eventModeAdapter =
ArrayAdapter.createFromResource(this,
                R.array.Event_Mode, android.R.layout.simple_spinner_item);
```

```
        spinnerEventMode.setAdapter(eventModeAdapter);

        ArrayAdapter<CharSequence> achievementsAdapter =
ArrayAdapter.createFromResource(this,
                R.array.Achievements, android.R.layout.simple_spinner_item);
        spinnerAchievements.setAdapter(achievementsAdapter);

        ivCalendarIcon.setOnClickListener(v -> showDatePicker());

        btnSubmit.setOnClickListener(v -> submitAchievement());

        fetchmethods();


    }

    private void fetchmethods() {

        if(currentUser==null) return;

        String teamLeaderName = currentUser.getDisplayName();


        // Query Projects to find the Special Lab where teamLeaderName ==
loggedInUserName
        Query query =
databaseReference.child("SpecialLab").child("Students").orderByChild("Name").e
qualTo(teamLeaderName);
        query.addListenerForSingleValueEvent(new ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {
                if (dataSnapshot.exists()) {
                    for (DataSnapshot snapshot : dataSnapshot.getChildren()) {
                        // Fetch the Special Lab for the logged-in user
                        specialLab = snapshot.child("Special
Lab").getValue(String.class);
                        String  Name =
snapshot.child("Name").getValue(String.class);
                        String RollNumber =
snapshot.child("ID").getValue(String.class);
                        String Year =
snapshot.child("Year").getValue(String.class);
                        String Department =
snapshot.child("Department").getValue(String.class);
                        etSpecialLab.setText(specialLab);
                        etName.setText(Name);
                        etRollNumber.setText(RollNumber);
                        etYear.setText(Year);
                        etDepartment.setText(Department);
```

```
                    etDepartment.setEnabled(false);
                    etSpecialLab.setEnabled(false);
                    etName.setEnabled(false);
                    etRollNumber.setEnabled(false);
                    etYear.setEnabled(false);
                }
            }
        }


        @Override
        public void onCancelled(@NonNull DatabaseError databaseError) {
            Toast.makeText(Achievementsadd.this, "Error fetching data.",
Toast.LENGTH_SHORT).show();
        }
    });
}

private void showDatePicker() {
    // Get current date
    Calendar calendar = Calendar.getInstance();
    int year = calendar.get(Calendar.YEAR);
    int month = calendar.get(Calendar.MONTH);
    int day = calendar.get(Calendar.DAY_OF_MONTH);

    // Show DatePickerDialog
    DatePickerDialog datePickerDialog = new
DatePickerDialog(Achievementsadd.this,
            (view, year1, month1, dayOfMonth) -> {
                // Format the selected date
                String selectedDate = dayOfMonth+ "-"+ (month1 +
1)+"-"+year1;
                tvSelectedDate.setText(selectedDate);
                tvSelectedDate.setEnabled(false);
            }, year, month, day);
    datePickerDialog.show();
}

private void submitAchievement() {
     name = etName.getText().toString();
     rollNumber = etRollNumber.getText().toString();
     year = etYear.getText().toString();
     department = etDepartment.getText().toString();
    String eventName = etEventName.getText().toString();
    String projectTitle = etProjectTitle.getText().toString();
    String proofLink = etProof.getText().toString();
    String eventDate = tvSelectedDate.getText().toString();
    String eventType = spinnerEventType.getSelectedItem().toString();
    String eventMode = spinnerEventMode.getSelectedItem().toString();
```

```
        String achievementType =
spinnerAchievements.getSelectedItem().toString();


        if (name.isEmpty() || rollNumber.isEmpty() || eventName.isEmpty() ||
projectTitle.isEmpty() || proofLink.isEmpty()) {
            Toast.makeText(Achievementsadd.this, "Please fill all the required
fields.", Toast.LENGTH_SHORT).show();
            return;
        }


        // Create Achievement object
        Achievement achievement = new Achievement(name, rollNumber, year,
department, specialLab, eventName, projectTitle,
                eventDate, eventType, eventMode, achievementType, "Pending",
proofLink);


        // Store in Firebase under Achievements/SpecialLab/RollNumber

databaseReference.child("Achievements").child(specialLab).child(rollNumber).se
tValue(achievement)
                .addOnCompleteListener(task -> {
                    if (task.isSuccessful()) {
                        Toast.makeText(Achievementsadd.this, "Achievement
submitted successfully.", Toast.LENGTH_SHORT).show();
                    } else {
                        Toast.makeText(Achievementsadd.this, "Error submitting
achievement.", Toast.LENGTH_SHORT).show();
                    }
                });
    }
}
```