Data Storage

Database	Local Storage
Better for larger amounts of data	Better for smaller amounts of data
Ideal for data shared between multiple users	Ideal for data specific to a single user
Data persists and is available across all devices a user logs in to	Data is tied to a single device or session; users will not be able to log in on other devices and retrieve their data
Limited by database storage capacity	Limited by device storage capacity
Ideal for security of user accounts and data	Not as secure for user accounts and data; ideal if we did not implement user accounts and passwords, only saved data and personal information locally

- Local storage of data would be much simpler to implement, however, it could be limited.
  - Java – Android SharedPreferences API for simple data
  - Java – SQLite (file-based database typically used for local storage) for complex data
- Database implementation would require either open source or paid resources.
  - MySQL – free for open source (Community Edition)

Local Storage Example:

Saving simple data (when user submits changes):

```
SharedPreferences prefs = getSharedPreferences("UserPrefs", MODE_PRIVATE);
SharedPreferences.Editor editor = prefs.edit();
editor.putString("name", userName);
editor.putFloat("height", height);
editor.putFloat("wingspan", wingspan);
editor.putString("difficulty", difficultyLevel);
editor.apply();
```

Retrieving simple data (when app starts):

```
SharedPreferences prefs = getSharedPreferences("UserPrefs", MODE_PRIVATE);
String userName = prefs.getString("name", "Unknown User");  // Default is "Unknown User"
float height = prefs.getFloat("height", 0);
float wingspan = prefs.getFloat("wingspan", 0);
String difficultyLevel = prefs.getString("difficulty", "beginner");  // Default is "beginner"
```

Saving complex data (when pathfinding completes):

```
public class ClimbingDBHelper extends SQLiteOpenHelper {

    private static final String DATABASE_NAME = "climbing.db";
    private static final String TABLE_PATHS = "paths";
    private static final String COLUMN_ID = "id";
    private static final String COLUMN_PATH = "path";
    private static final String COLUMN_NAME = "name"; //possibly timestamp

    public ClimbingDBHelper(Context context) {
        super(context, DATABASE_NAME, null, 1);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        // Create the paths table
        db.execSQL("CREATE TABLE " + TABLE_PATHS + " (" +
                COLUMN_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
                COLUMN_PATH + " TEXT, " +
                COLUMN_NAME + " TEXT)");
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_PATHS);
        onCreate(db);
    }

    // Save a path
    public void savePath(String path, String pathName) {
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put(COLUMN_PATH, path);
        values.put(COLUMN_NAME, pathName);
        db.insert(TABLE_PATHS, null, values);
    }
```

Retrieving complex data (when user views/selects a path from history):

```
    // Retrieve a path by its name
    public String getPathByName(String pathName) {
        SQLiteDatabase db = this.getReadableDatabase();
        Cursor cursor = db.query(TABLE_PATHS,
                new String[]{COLUMN_PATH},
                COLUMN_NAME + "=?",
                new String[]{pathName},
                null, null, null);

        if (cursor != null && cursor.moveToFirst()) {
            String path = cursor.getString(0);
            cursor.close();
            return path;
        }
        return null;  // Return null
    }

    // Retrieve all paths
    public List<String> getPaths() {
        List<String> paths = new ArrayList<>();
        SQLiteDatabase db = this.getReadableDatabase();
        Cursor cursor = db.rawQuery("SELECT * FROM " + TABLE_PATHS, null);
        if (cursor.moveToFirst()) {
            do {
                paths.add(cursor.getString(cursor.getColumnIndex(COLUMN_PATH)));
            } while (cursor.moveToNext());
        }
        cursor.close();
        return paths;
    }
}
```

Resources:

https://developer.android.com/training/data-storage/shared-preferences

https://developer.android.com/training/data-storage/sqlite#java