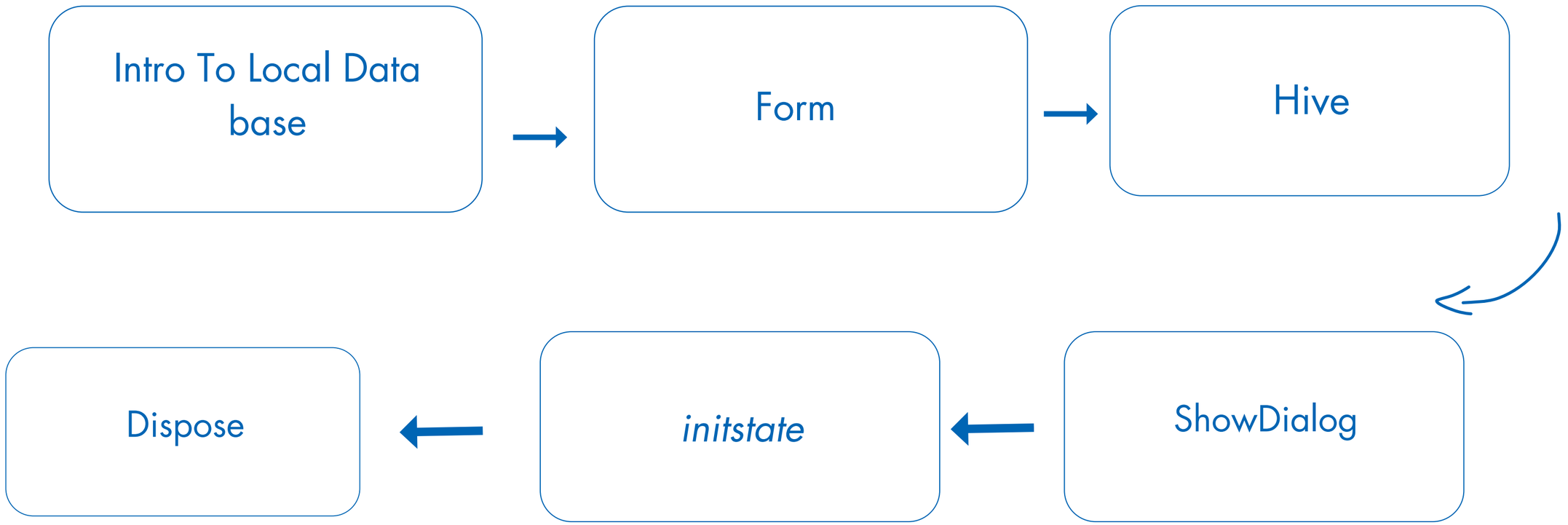
MACHINE LEARNING

SESSION 10

[illegible]


Intro To Local Data base
Form
Hive
ShowDialog
initstate
Dispose

NTI LINE

A local database stores data on the user’s device, allowing the app to:

- Work offline
- Retain user data after closing the app
- Quickly read/write small to medium-sized data

One of The Most Popular Local Databases in Flutter is ***Hive***

Building forms for user input

- **Form:** groups input fields.
- **TextFormField:** supports built-in validation.

```dart
final _formKey = GlobalKey<FormState>();

Form(
  key: _formKey,
  child: TextFormField(
    validator: (value) => value!.isEmpty ? 'Required' : nul
l,),
);
```

Dialog

Showing messages or confirmations

Used to ask for confirmation before actions (like delete or exit).

```dart
showDialog(
  context: context,
  builder: (_) => AlertDialog(
    title: Text('Delete Note?'),
    content: Text('Are you sure?'),
    actions: [
      TextButton(onPressed: () => Navigator.pop(context), child: Text('Cancel')), TextButton(onPressed: () {
        Hive.box('notes').delete('myNote');
        Navigator.pop(context);
      }, child: Text('Yes')),
    ],
  ),
);
```

HIVE

What is Hive?

- A lightweight & blazing fast NoSQL database.
- Stores data locally on the user's device.
- Works offline.
- Easy to integrate into Flutter apps.

NoSQL stands for "Not Only SQL".

It refers to non-relational databases that do not use traditional tables or schemas like SQL databases.

HIVE

Hive Step by Step

1-Add Hive Dependencies

```yaml
dependencies:
  flutter:
    sdk: flutter
  hive: ^2.2.3
  hive_flutter: ^1.1.0

dev_dependencies:
  hive_generator: ^2.0.1
  build_runner: ^2.4.5
```

HIVE

2 – Create a Model Class

```dart
import 'package:hive/hive.dart';
import 'package:hive_flutter/hive_flutter.dart';

part 'note_model.g.dart';

@HiveType(typeId: 0)
class NoteModel extends HiveObject {
  @HiveField(0)
  final String title;
  @HiveField(1)
  final String body;
  @HiveField(2)
  final String date;
  NoteModel({required this.title, required this.body, required this.dat
});
```

HIVE

3 – Initialize Hive

openBox **must be done before accessing the box.**

```dart
void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Hive.initFlutter();
  await Hive.openBox("notes");
  runApp(Noteapp());
}
```

4 – Open a Box

```
1   var box = Hive.box('note
    s');
```

A box is like a local table or file to store data.

HIVE

5– Save Data to Hive

```
box.put('title', 'My First Note');
```

put(key, value) : both key and value can be any type.

HIVE

6 – Read Data from Hive

```dart
var note = box.get('title');
print(note); // Output: My First Note
```

Returns null if the key doesn’t exist.

NOTE

initState (Widget initialization)

- Called before the widget builds.
- Good place to load data from Hive.

```dart
@override
void initState() {
  super.initState();loadData();
}
```

dispose

Clean up resources
Required when using controllers, focus nodes, or streams.

Prevents memory leaks.

```dart
@override
void dispose() {
  _controller.dispose();
  super.dispose();
}
```

IEEE | FCI - Helwan Student Branch

THANK YOU

SEE YOU NEXT TIME

S25' Training Sessions