Лабораторийн тайлан №15

Flutter фрэймворкын үндсэн ойлголт ба түүний хэрэглээ

1. MediaQuery - дэлгэжийн хэмжээ дэлгэцэнд үзүүлэх тохиргоо гэх мэт одоогийн мэдээллийг өгдөг класс.  
   ***Жишээ нь:*** *MediaQuery.of(context).size, дэлгэцийн хэмжээ MediaQuery.of(context).orientation,одоо байгаа байрлал MediaQuery.of(context).textScaleFactor дэлгэцийн scale хийгдсэн утга* ***гэх мэт….  
   Мөн*** *тухайн widget эвдрээд байвал MaterialApp – даа тохируулж болно.****Жишээ нь:****Хэрэглээний хувьд* ***builder: (context, child) => MediaQuery(data: MediaQuery.of(context).copywith(textScaleFactor: 1.0), child: child!))*** *Системийн scaleFactor нэмэгдсэн ч гэсэн аппын хэмжээ яг 1 ээрээ байна.*
2. Semantic – Дэлгэцийн агуулгыг уншиж өгөх боломжийг олгодог widget.   
   ***Хэрэглээ:******return SemaOrdinalSortKey(Ontics(  
   sortkey:),  
   header: true,  
   child:Text(“myText”)),);***
3. GlobalKey ашиглан context – д хандах  
   Аппликешны хувьд бүхэлдээ хүчинтэй давтагдахгүй байх түлхүүр.Үүнийг ямар нэг widget – ийн build() функцийн гаднаас тухайн widget – ийн төлөвт хандах удирдахад ашиглаж болно.Тухайн context – ийг хаанаас ч хандах боломжтой байхаар ашиглахын тулд хэрэгжүүлдэг. GlobalKey нь **UI-ийн тодорхой хэсгийн state руу хандах** богино хугацааны шийдэл.  
   ***Хэрэглээ:  
   static final navigatorKey = GlobalKey<NavigatorState>();  
   GlobalKeys.dart  
   class GlobalKey{  
   static final navigatorKey = GlobalKey<NavigatorState>();  
     
   }  
   main.dart  
   MaterialApp дотроо:  
    navigatorKey: GlobalKeys.navigatorKey***
4. Translation – Нийт аппын хувьд нийт дэлгэцийн өгөгдөл бусад зүйлсийг GlobalKey – ээрээ аваад нийт дэлгэцэнд байгаа зүйлсээс яг тийм json – д тодорхойлсон өгөгдөл нэршил байвал түүний өөрчлөх байдлаар хэрэгжүүлдэг.Хэрэгжүүлэхдээ тухайн **label “shop”.tr()** хөрвүүлнэ гэдгийг илэрхийлээд хэрэгжүүлдэг.
5. FuturBuilder – Ассинхрон функцын үед ui – ийн ба widget – ийг өөрөө build хийж өгөхөд ашигладаг.Жишээ нь өгөгдлийн сангаас өгөгдлүүд авах гэх мэт.Энэ нь мөн л widget.  
   **future:** өгөгдөл ачаалах зэрэг асинхрон функц авна.  
   **builder:** асинхрон функцыг шалгаж widget хэрхэн байгуулахыг зааж өгнө.  
   **snapshot.connectionState:** ямар төлөвт байгааг заана.  
   **snapshot.hasData:** Өгөгдөл байгаа эсэхийг шалгадаг.  
   **snapshot.data:** өгөгдөлийн утгыг авна.Snapsot дээр тухайн future – ийн өгөгдлүүд болон бусад төлвүүдийг хадгалдаг.
6. SharedPreferences – Үндсэн өгөгдлийн төлүүдийг хадгалахын тулд ашигладаг.***FlutterSecureStorage()*** энэ нь тухайн token – г encryption хийгээд хадгалдаг.
7. **Post request** - **POST /auth/login яагаад token буцаадаг вэ?** Энэ API-ийн /auth/login endpoint нь:**таны илгээсэн хэрэглэгчийн нэр ба нууц үгийг шалгаадамжилттай нэвтэрсэн бол token (жишээ нь JWT)** буцаадаг.

**☑️ Яагаад POST ашигладаг вэ?**Хэрэглэгчийн username ба password гэх мэт **эмзэг өгөгдөл**-ийг body хэсэгт илгээдэг.GET хүсэлтээр бол ийм мэдээлэл URL дээр харагддаг тул **аюулгүй бус**.Нэвтрэх нь **системийн төлөвийг өөрчилдөг** үйлдэл (authentication state нэмэгддэг).Иймээс **POST нь тохиромжтой сонголт** юм.

1. **Token -** token гэж юу вэ?Энэ нь таныг амжилттай нэвтэрсэн гэдгийг батлах дижитал тэмдэг юм. Дараагийн хүсэлтүүддээ Authorization header-ээр дамжуулан илгээнэ:
2. **EasyLoading –** Дэлгэц уншиж буй хэсгийг харуулахын тулд ашигладаг.
3. **UUID –** Автоматаар ID үүсгэж өгдөг.Буцаах утга string байна.
4. **ImagePicker –** Файлд хадгалахын тулд зөвшөөрөл хэрэгтэй.Тухайн зөвшөөрлийг олгохын тулд ашигладаг.жишээ нь camera нээгдэх үе асуухад зөвшөөрөл олгодог.IOS – ийн хувьд.
5. **PickedImage –** XFILE буцдаг тул түүнийг энгийн файл болгон хөрвүүлэх ёстой.
6. **PathProvider –** Аппын суусан path – уудыг өгдөг.**path** нь url – ийг нийлүүлэхэд хялбар байх боломжийг олгодог.    
   ***final appDir = await syspath.getApplicationDocumentsDirectory();***

***final filename = path.basename(a.image.path); төхөөрөмжийн зураг хадгалах directory – г авна мөн тухайн параметрээр ирж буй зургын нэрийг мөн авна.***

1. **Location –** Газрын байршилыг авдаг сан.