Projektdokumentation, Gruppe 5 - SKS Booking

Om

Dette projekt består af en frontend-applikation i Flutter og en backend-applikation i .NET til at stå for brugeroplevelsen og funktionaliteten af SKS Bookings service. Billeder uploades til og opbevares i en S3 bucket. Frontenden er testet i Windows, Android og web (Google Chrome), både med lokal og online backend.

Det fulde repository kan findes her:
https://github.com/Mercantec-GHC/h4-projekt-gruppe-5-1/tree/main

Kendte problemer

App

- OpenStreetMaps API virker ikke, når Flutter startes som Windows-applikation. Dette skyldes, at platformen ikke er understøttet. Dette kan vi ikke fikse

- Delete kald til SecureStorage sletter ikke altid den relevante data, når Flutter startes som Windows-applikation, og kan medføre, at brugertypen, samt en token (forældet og ugyldig efter ~30 min), kan overleve forskellige instanser indtil nyt login. Hvis den gemte token er for gammel, er sendt data inkongruent. Dette sker fordi typen påstår, du er logget ind, men din token er ugyldig til verificering, og kræver derfor et frisk login

Delt

- Det er pt ikke muligt at ændre i et lejeboligopslag, se bookingoversigt eller ændre en booking. De dele nåede vi ikke

Før start

Hvis API'en opsættes lokalt, skal den være konfigureret og opstartes før Flutter-applikationen (se Konfiguration).

Hvis endpoints til API og andre services ændres, skal de reflekteres i Flutter-applikationen (se Konfiguration).

Database og diagram

Der bliver gjort brug af 3 forskellige tabeller i vores projekt: User, Rental og Booking.

User

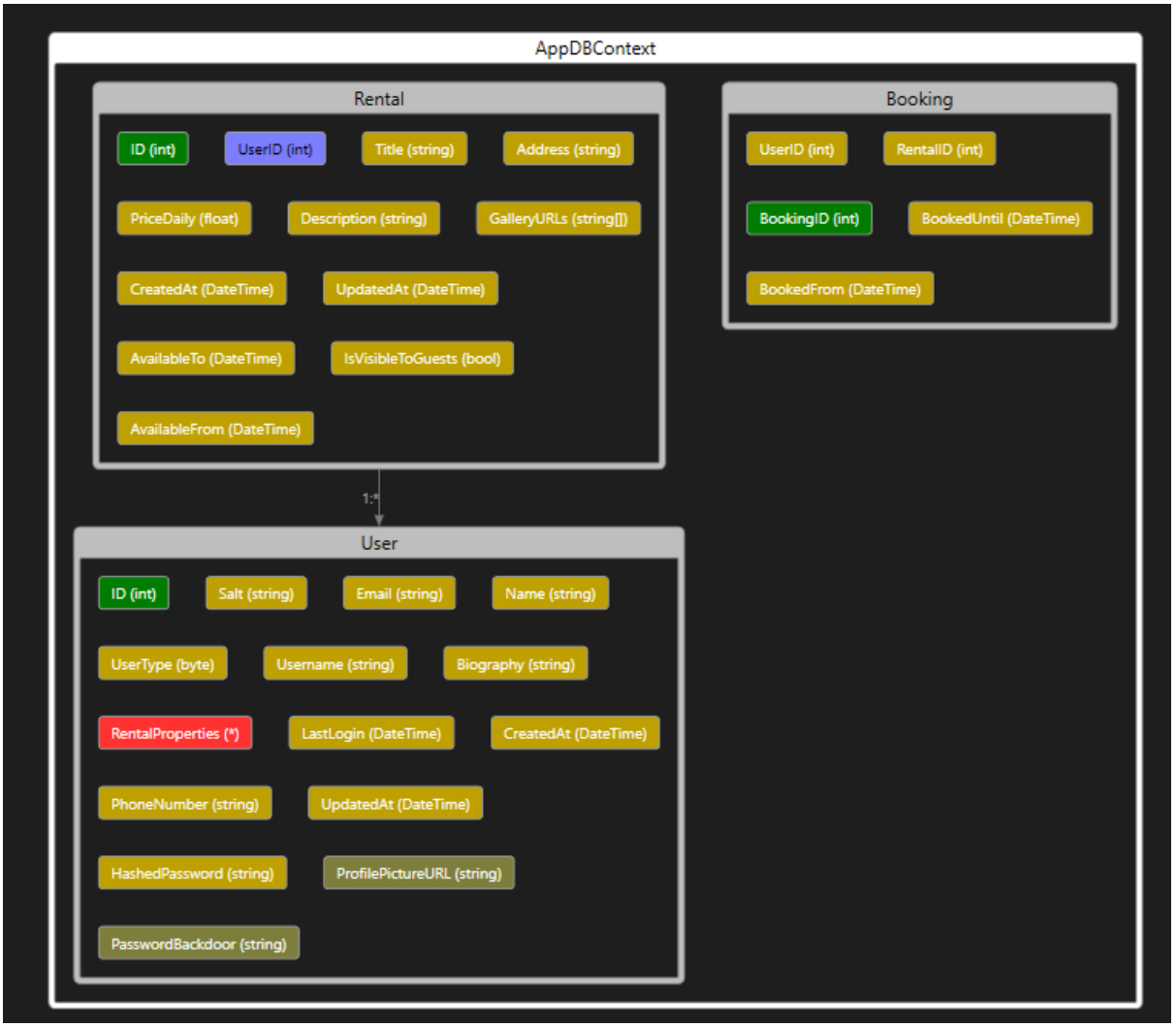
User er vores brugere på platformen. De har en brugertype, som står for privilegier og status, dvs om den er almindelig bruger, udlejer eller administrator. Derudover holder de informationer til at udfylde en brugerprofil, og ikke-almindelige brugere har også mulighed for at have associerede lejeboliger, som de udlejer.

Rental

Rental er vores tabel for lejeboliger. Lejeboligerne har en kort, beskrivende titel, en adresse, pris, beskrivelse, og et galleri af billeder. De kan skjules i søgning fra folk, der ikke er loggede ind, og har en reference til hvem, der ejer den.

Booking

Bookingtabellen er betydeligt mere simpel. Den tilknytter en bruger og en lejebolig i en given periode, og tillader, at det kan ske flere gange mellem den samme bruger og bolig.



Pakker/Package Management

Vores projekt gør brug af følgende pakker og version:

API

AWSSDK.S3	3.7.402.4
BCrypt.Net-Next	4.0.3
Microsoft.EntityFrameworkCore	8.0.8
Microsoft.EntityFrameworkCore.Tools	8.0.8
Microsoft.AspNetCore.Authentication.JwtBearer	8.0.8
Swashbuckle.AspNetCore	6.7.3
Npgsql.EntityFrameworkCore.PostgreSQL	8.0.4

App

provider	6.0.0
http	1.2.2
intl	0.19.0
flutter_secure_storage	8.1.0
flutter_map	7.0.2
latlong2	0.9.1
location	6.0.2
expandable_text	2.3.0
permission_handler	11.3.1
image_input	0.0.5
dio	5.7.0
file_picker	8.1.2
shared_preferences	2.3.2

Konfiguration

API

S3 Service Endpoint

Adressen til brugt S3 service + bucket er defineret i /Service. Bliver der brugt et andet endpoint, skal selve servicens “ServiceURL” rettes. Derudover skal begge metoder rettes til ved “bucketname” og “key”. Den første metode skal også have “imageURL” rettet.

```
3 references
public async Task<string> UploadToS3(Stream fileStream, string uid, ImageDirectoryType type) {
    var request = new PutObjectRequest {
        InputStream = fileStream,
        BucketName = "sks",
        Key = $"{type}/{uid}.png",
        DisablePayloadSigning = true
    };

    var response = await _s3Client.PutObjectAsync(request);

    if (response.HttpStatusCode != System.Net.HttpStatusCode.OK) {
        throw new AmazonS3Exception($"Error uploading file to S3. HTTP Status Code: {response.HttpStatusCode}");
    }

    var imageUrl = $"https://sks.mercantec.tech/sks/{type}/{uid}.png";
    return imageUrl;
}
```

Secrets (lokalt)

Relevant lokal konfiguration af API findes i appsettings.json i SKSBookingAPI-mappen. Her indtastes relevante nøgler:

JSON Web Tokens

```
"JwtSettings": {
    "Issuer": "",
    "Audience": "",
    "Key": ""
}
```

Database

```
"ConnectionStrings": {
    "DefaultConnection": ""
}
```

S3 Service

```
"S3ServiceSettings": {
    "AccessKey": "",
    "SecretKey": ""
}
```

App

Endpoint URL

Flutter-applikationen forbinder til roden givet som “baseUrl” i main.dart. Ændring af API-adresse skal reflekteres her (i dette tilfælde, erstat https://localhost:7014/api):

```
45  class MyAppState extends ChangeNotifier {
46    final ApiService apiService =
47        ApiService(baseUrl: 'https://localhost:7014/api');
48    late var success = false;
```

Filstruktur

Docs

Indeholder logbøger, kravspecifikation og projektpitch ift kundeopgaven fra gruppe 4. Derudover findes vores egen case givet til gruppe 6.

FlutterApp

Indeholde vores Flutter frontend-applikation. Koden findes i /lib, og udover filerne main.dart og api.dart findes resten delt op i to indre mapper; “models” og “pages”.

/models
Filerne her er ansvarlige for modellering af data i vores frontend (dvs data ind i Flutter applikationen), når det kommer fra vores backends endpoints.

/pages
Filerne her er alle de forskellige sider, man kan støde på i vores app. De indeholder generelt sidestruktur og funktionalitet i isolation, men kan godt lave kald på hinanden via fx Navigator.

SKSBookingAPI

Indeholder vores .NET og EF backend, som er ansvarlig for vores API endpoints, database, håndtering af requests og styring, mm. Opsætningen findes i Program.cs, og diverse nøgler til fx JSON Web Tokens og vores S3 service bucket gives i appsettings.json.

De vigtigste komponenter er delt op således:

/Context
Indeholder vores databasekontekst.

/Controllers
Her findes controllere til de forskellige endpoints. De står for at håndtere endpoints og de requests, der bliver sendt dertil, inklusivt at være bindeled med hensyn til datakonvertering og kommunikation med vores S3 service.

/Models
Datamodellerne til vores database og DTO’er til controllere. De definerer, hvordan de forskellige tabeller i vores database ser ud, og DTO’erne hvilke data kommer ind og ud af vores backend.

/Service
Opsætning af og tilkobling til vores S3 service, som står for upload (og potentielt sletning) af billeder, som kan hentes via URL til vores frontend-applikation.