

UAS MOBILE PROGRAMMING



Anggota :

Dennis Joel (825210023)

Ferry Wilson (825210017)

Jason (825210103)

**PROGRAM STUDI SISTEM INFORMASI
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS TARUMANAGARA**

A. TEAM SCRUM

1. Product Owner memiliki peran dalam Scrum yang bertanggung jawab untuk mengelola Product Backlog, menentukan prioritas, dan memastikan pengembangan produk sesuai dengan kebutuhan dan ekspektasi pengguna serta pemangku kepentingan.

Nama Product Owner :

- Harbet Christian

2. Scrum Master memiliki peran dalam Scrum yang bertanggung jawab memfasilitasi dan mendukung tim Scrum dalam menerapkan prinsip-prinsip dan praktik Scrum, serta menghilangkan hambatan yang menghalangi produktivitas tim.

Nama Scrum Master :

- Jason

3. Tim Developer memiliki peran dalam Scrum adalah kelompok individu yang bertanggung jawab untuk merancang, mengembangkan, dan menguji produk selama Sprint. Mereka bekerja secara kolaboratif dan otonom untuk mencapai tujuan Sprint.

Nama Scrum Master :

- Ferry Wilson
- Dennis Joel

B. SPRINT BACKLOG

Sprint Backlog adalah daftar konkret item pekerjaan yang dipilih oleh Tim Developer selama Sprint dan berisi detail pekerjaan, estimasi waktu, dan tanggung jawab individu. Berikut adalah Sprint Backlog untuk pengembangan aplikasi yang bernama “Resepku” merupakan suatu aplikasi yang berisi tentang resep makanan.

SPRINT 0 (PERSIAPAN)

Hari 1 :

- Team Building
- Diskusi untuk menemukan praktik dasar dan cara kerja dalam Scrum

Hari 2 :

- Team mendefinisikan bagaimana mereka akan bekerja : memilih tools dan aturan
 - JIRA
 - Definition of Done
 - Definition of Ready

- Sprint Duration : 2 Weeks
- Daily Scrum : 9.30 am di ruang meeting perusahaan

Hari 3 :

- Product Backlog

1. Sign up dan Sign in aplikasi (**5 poin**)

User Story :

Sebagai user , saya tidak memiliki akun maka saya harus mendaftar terlebih dahulu (Sign up) atau jika saya sudah memiliki akun sehingga saya bisa login ke dalam aplikasi (Sign in)

2. User dapat melihat tampilan main page (**8 poin**)

User Story :

Sebagai user, saya ingin melihat berbagai fitur yang ada di aplikasi sehingga saya dapat memilih fitur yang saya inginkan.

3. User dapat melihat resep makanan (**5 poin**)

User Story :

Sebagai user , saya ingin dapat melihat beberapa resep makanan yang tersedia sehingga saya dapat mengikuti resep masakan beserta tutorialnya

4. User dapat memilih category resep makanan (**8 poin**)

User Story :

Sebagai user, saya ingin mencari resep makanan sesuai kategori contohnya nasi sehingga saya bisa melihat berbagai resep makanan dengan menggunakan nasi

5. User dapat mencari resep makanan (**5 poin**)

User Story :

Sebagai user, saya ingin mencari resep makanan dengan judul contohnya seperti ayam rendang sehingga saya bisa melihat resep makanan tersebut dan mengikuti tutorial memasak

6. User dapat melihat profile account (**4 poin**)

User Story :

Sebagai user, saya ingin melihat profile saya pada aplikasi sehingga saya dapat melihat data-data pribadi saya yang ditampilkan di profile

7. User dapat mengupload resep (**8 poin**)

User Story :

Sebagai user, saya ingin mengupload berbagai resep makanan sehingga saya bisa menunjukan resep makanan saya kepada user lain

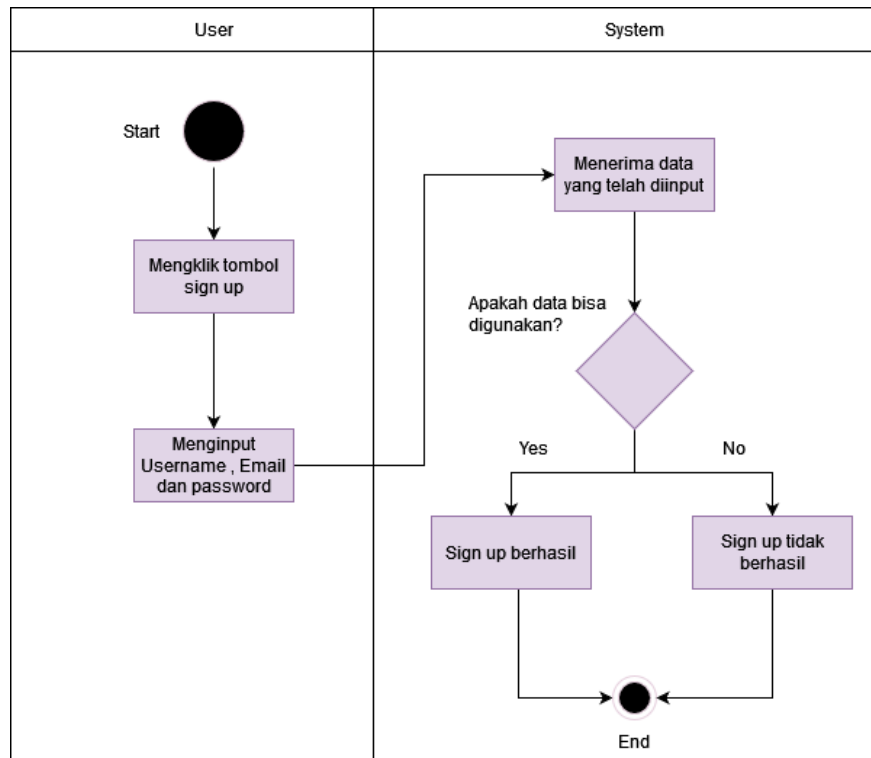
8. User dapat membuka settings (**5 poin**)

User Story :

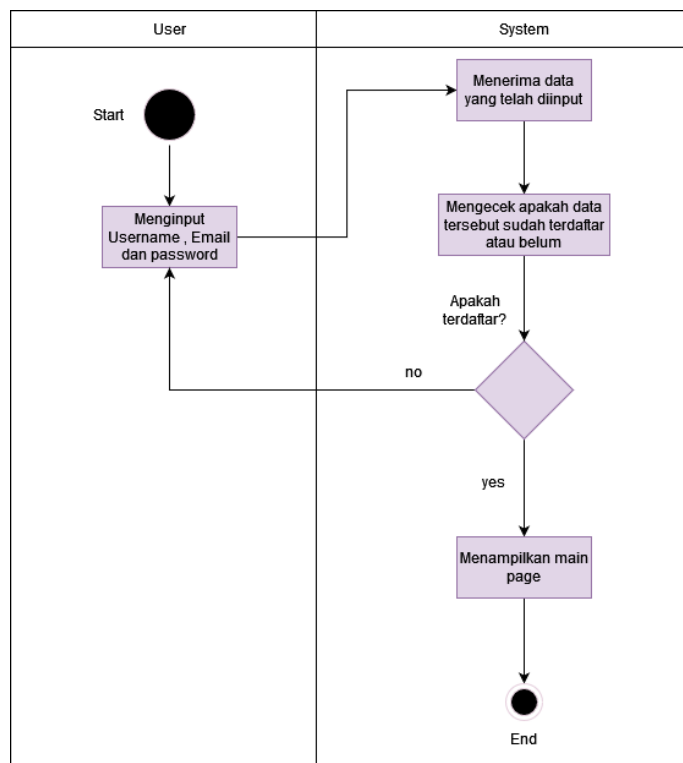
Sebagai user, saya ingin mengubah data-data pribadi saya seperti password sehingga akun saya akan jauh lebih aman

- UML
 - Activity Diagram

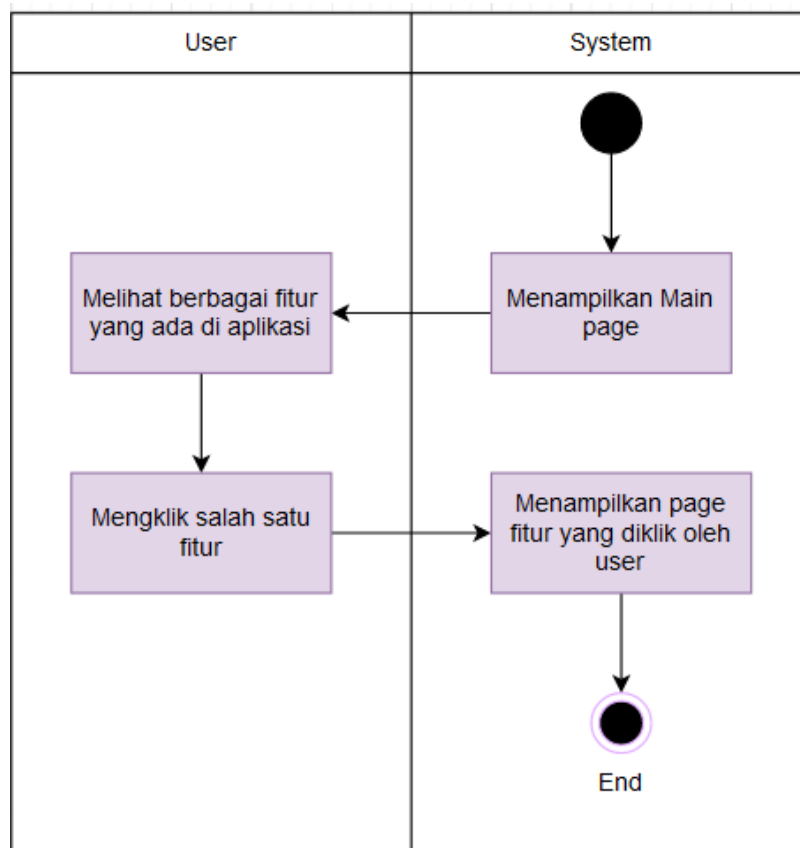
SIGN UP



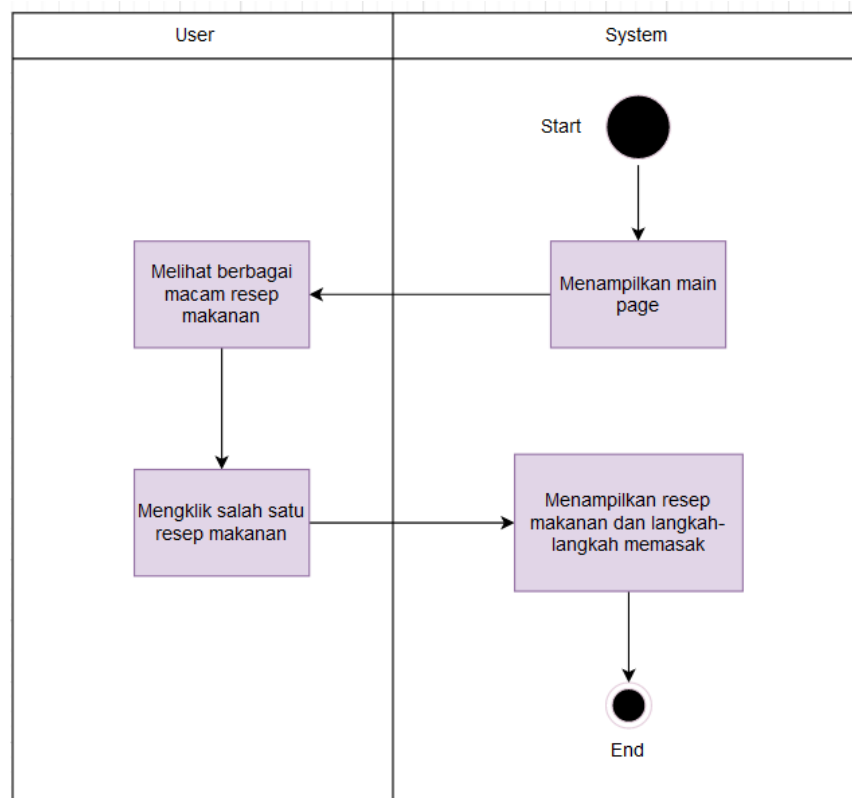
SIGN IN



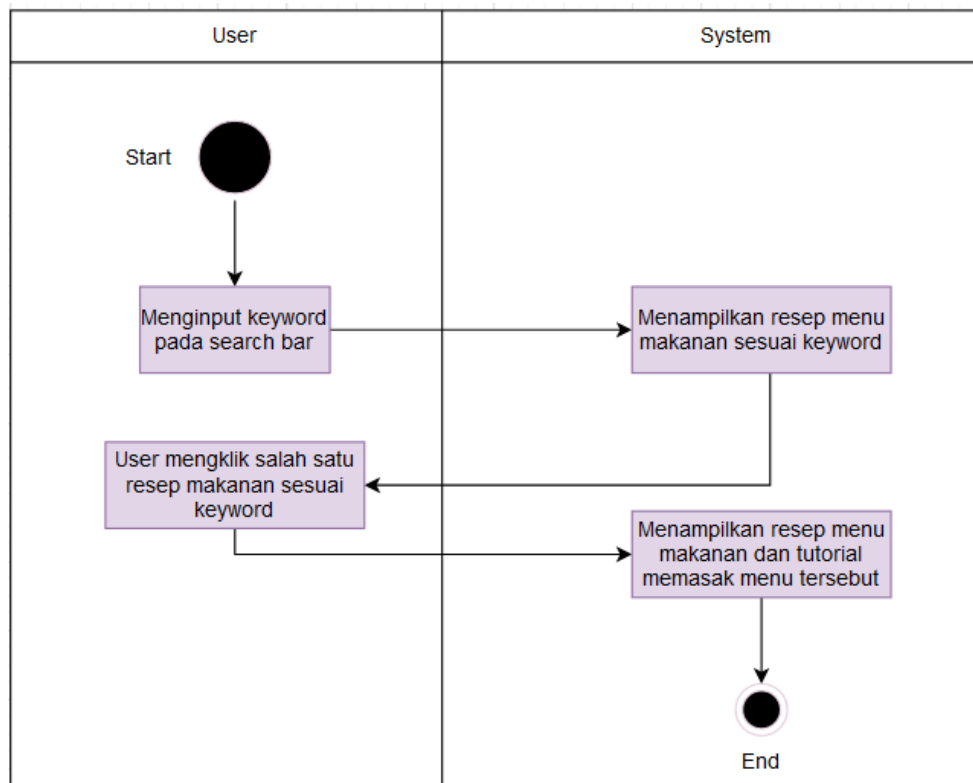
MAIN PAGE APLIKASI



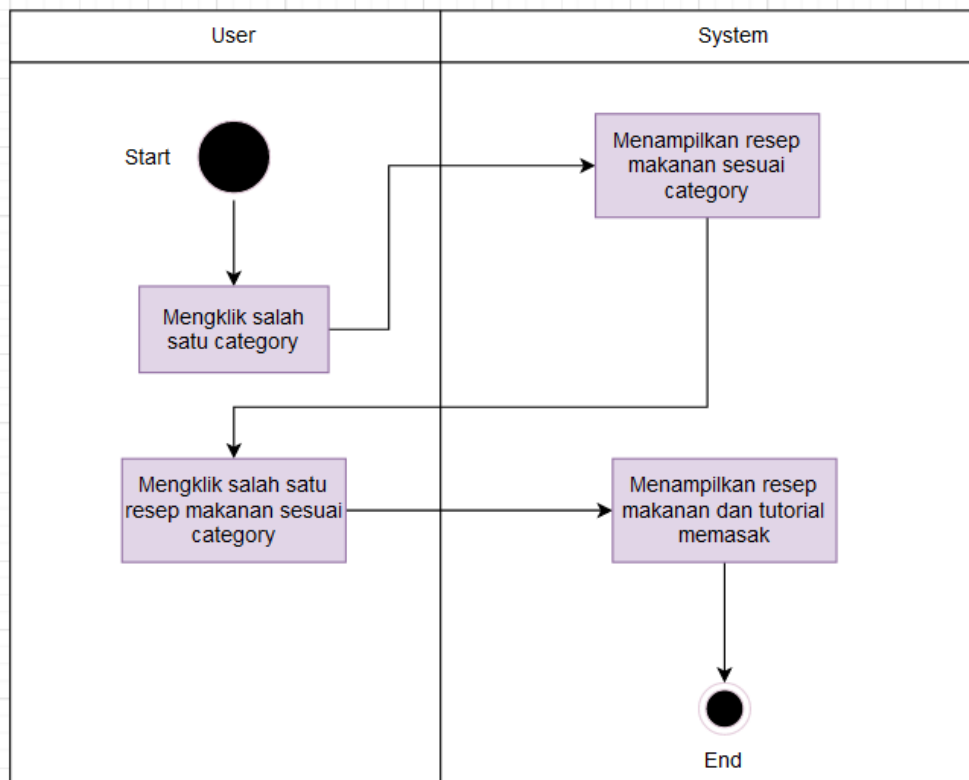
PAGE RESEP APLIKASI



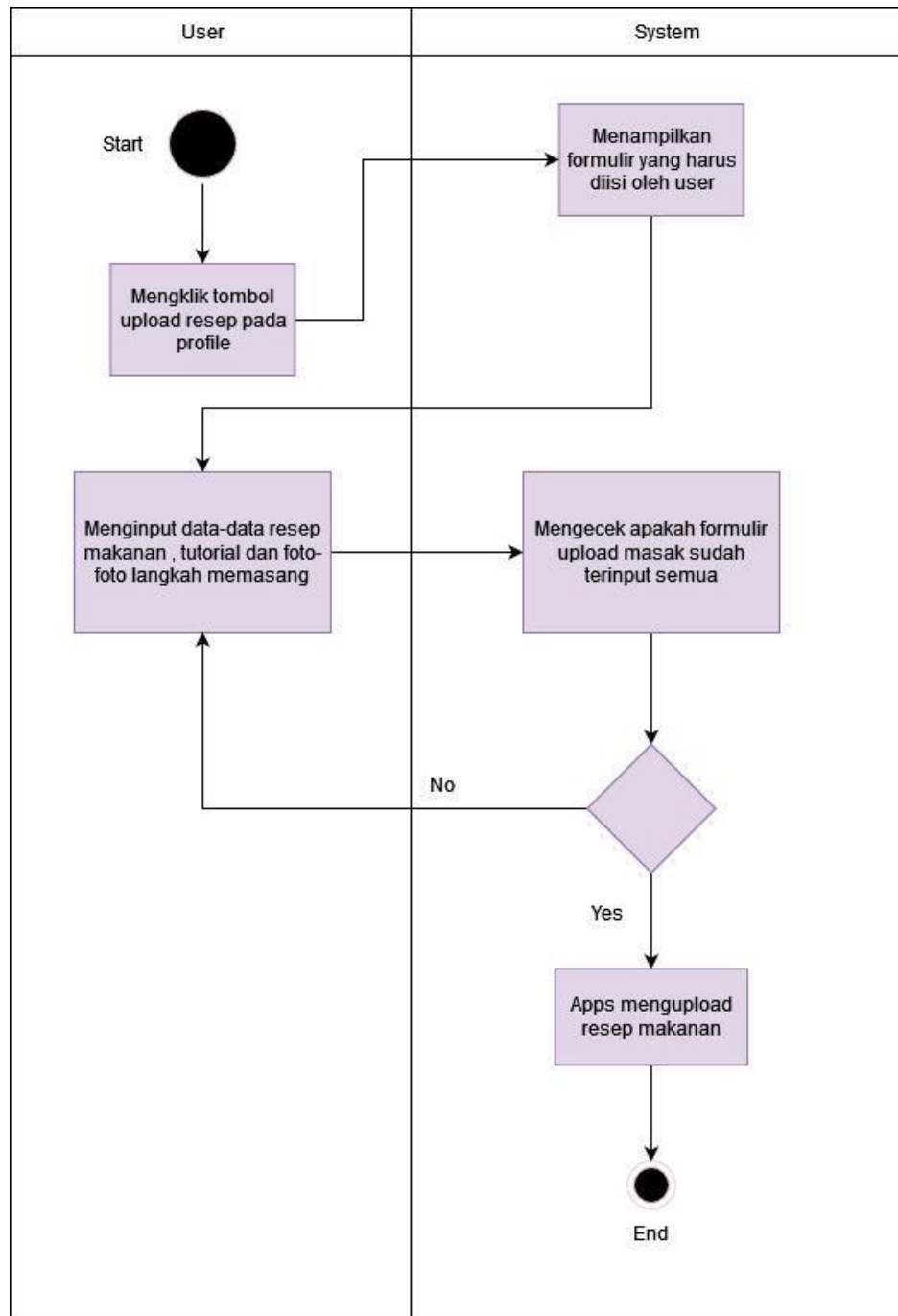
SEARCH BAR



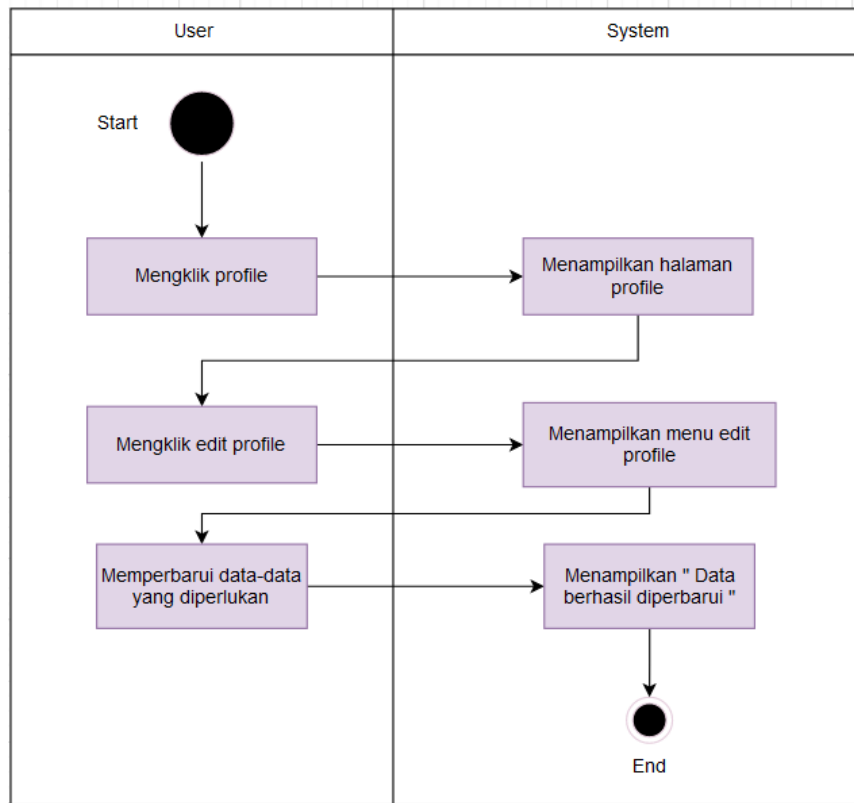
CATEGORY



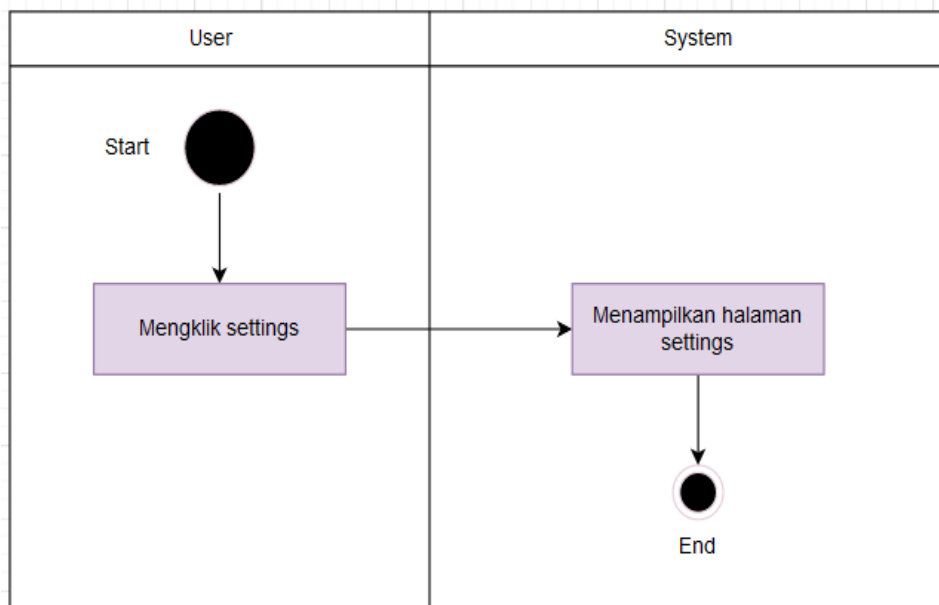
UPLOAD RESEP MAKANAN



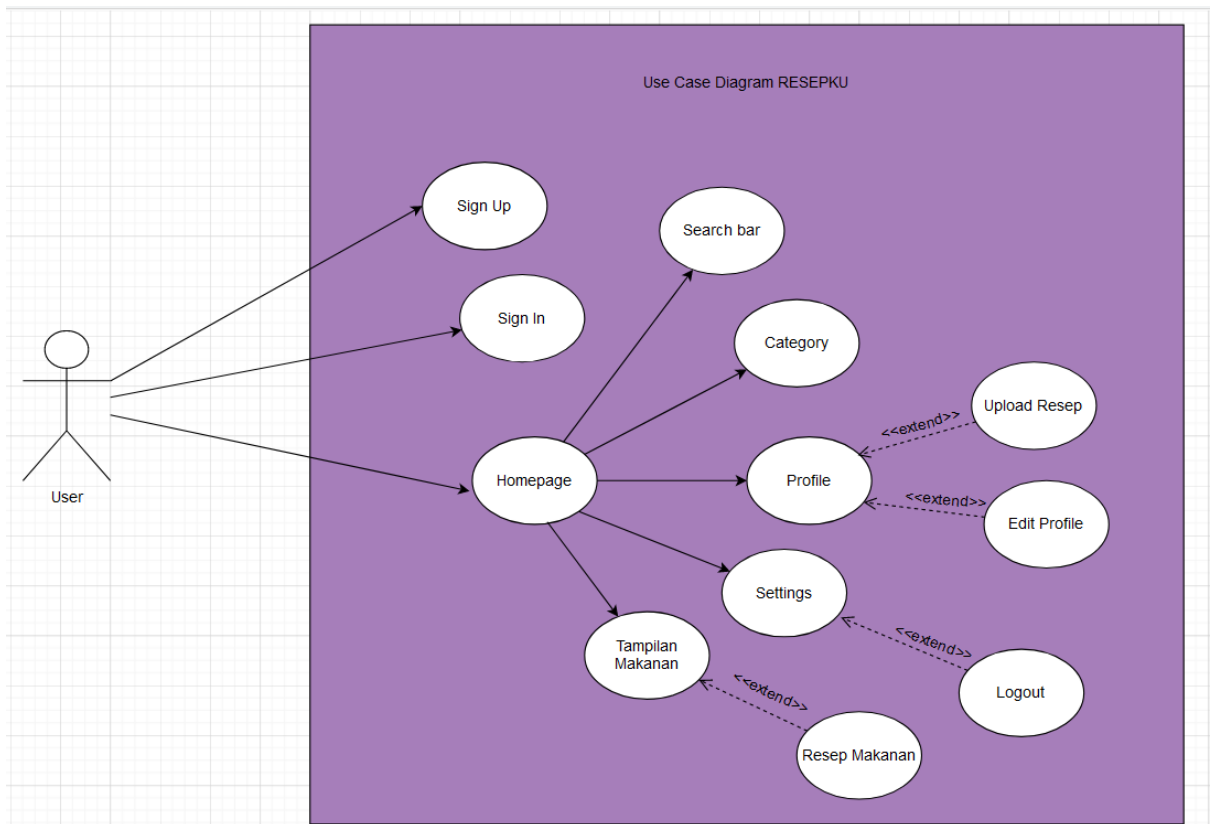
EDIT PROFILE



SETTINGS APLIKASI



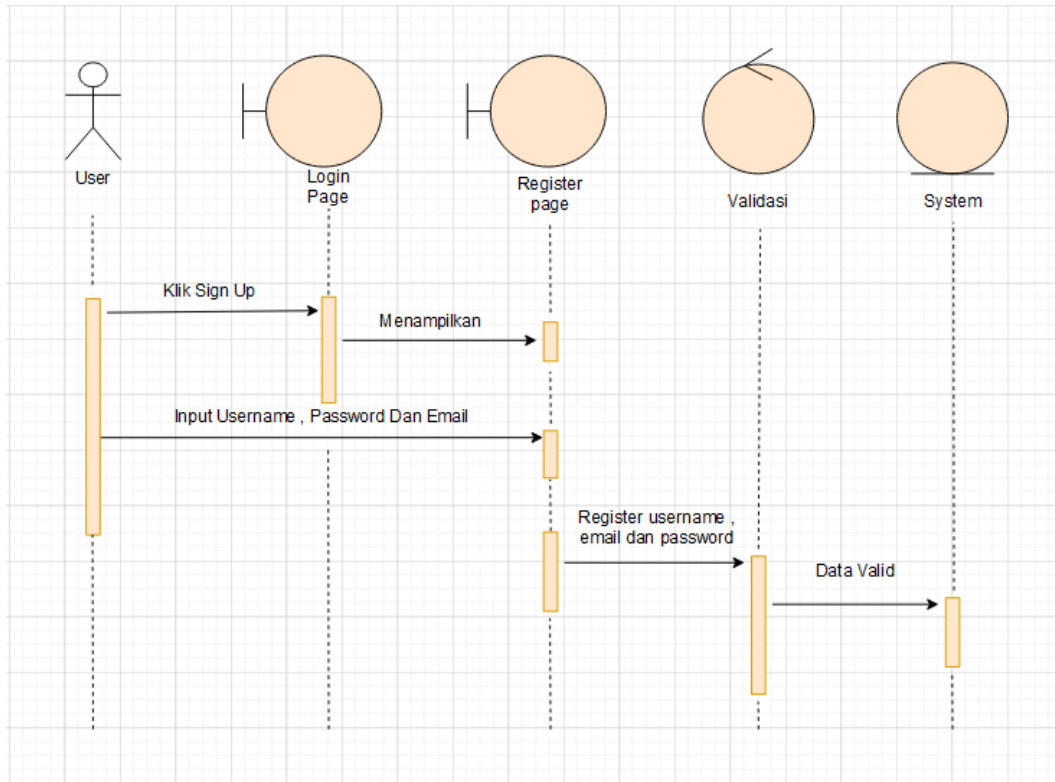
○ Use Case Diagram



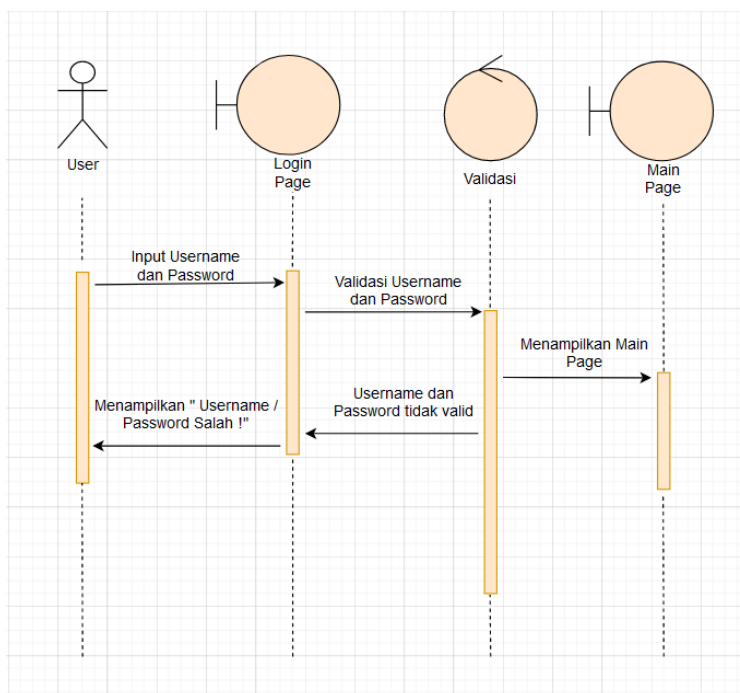
Berikut adalah proses use case diagram pada aplikasi RESEPKU. Jika user tidak memiliki akun maka user wajib untuk melakukan sign up. Jika user sudah memiliki akun maka user hanya perlu untuk sign in ke dalam apps. Setelah sign in apps akan menampilkan homepage yang berisi search bar , kategori , profile , settings dan tampilan makanan. Pada Search bar user dapat menginput kata kunci dan setelah user menginput kata kunci apps akan menampilkan resep makanan berdasarkan kata kunci contohnya seperti ayam goreng atau ikan goreng Pada Category user dapat memilih category makanan yang ada di apps contohnya seperti nasi atau mie. Pada profile memiliki fitur upload resep dan edit profile. User dapat mengupload resep makanan sendiri dan juga dapat mengedit profile akun user. Pada Tampilan makanan yang berada di homepage user dapat mengklik salah satu menu makanan dan apps akan menampilkan resep makanan tersebut. Pada Settings user dapat melakukan logout pada aplikasi.

- **Sequence Diagram**

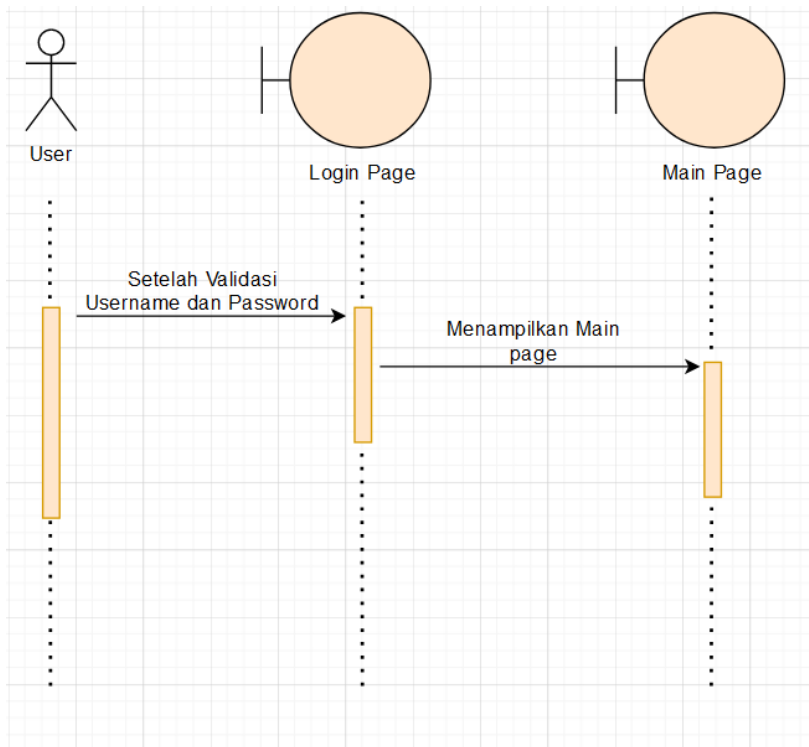
SIGN UP



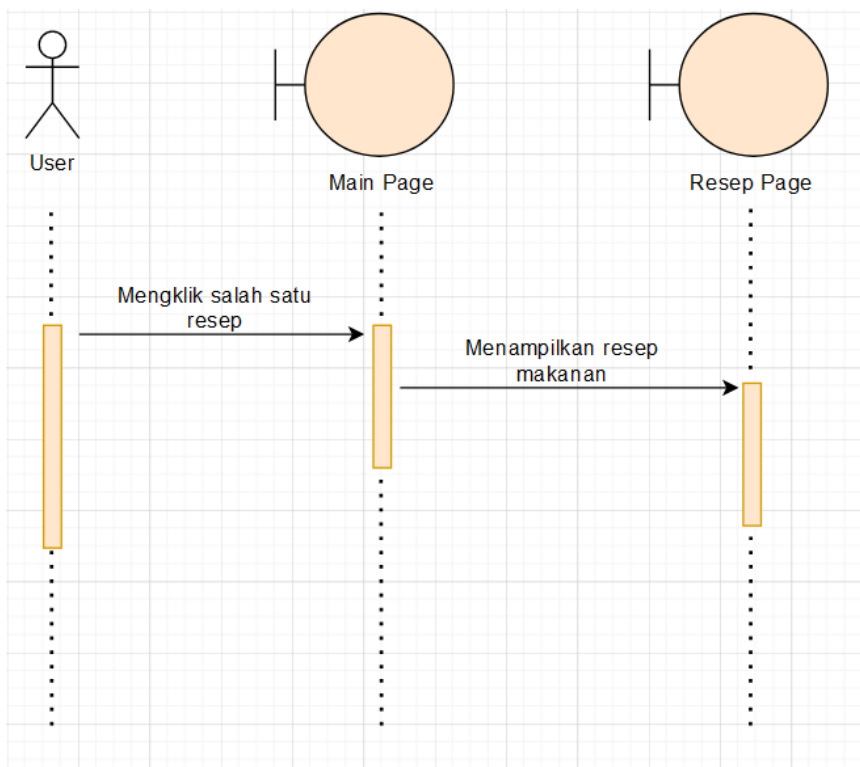
SIGN IN



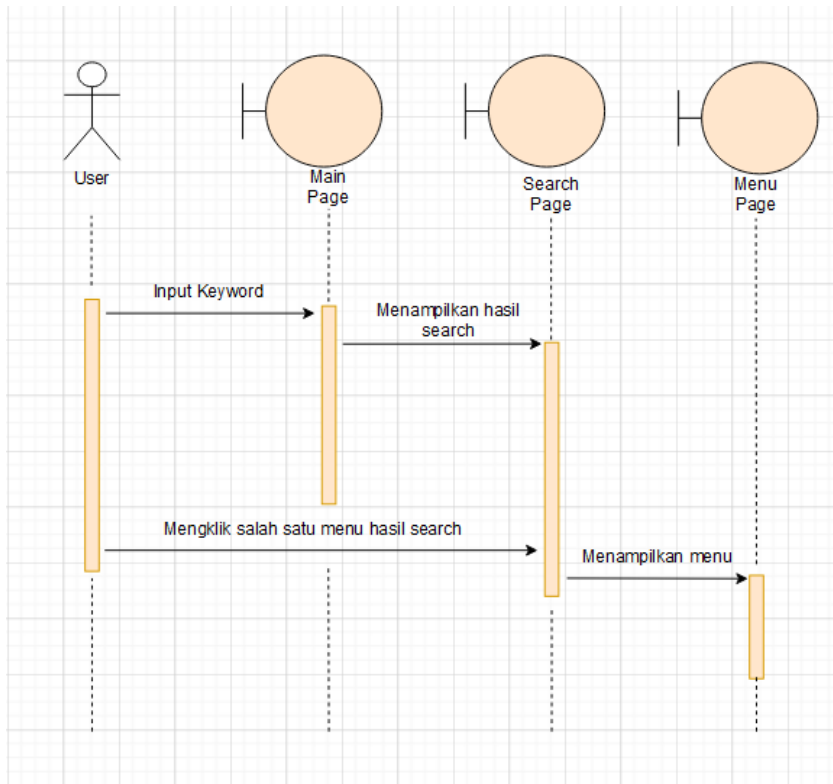
MAIN PAGE APLIKASI



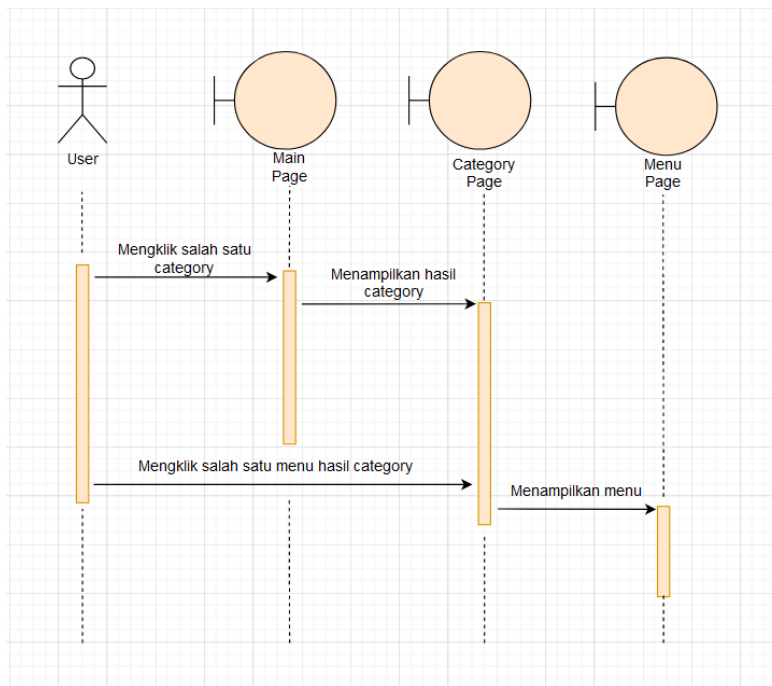
RESEP PAGE APLIKASI



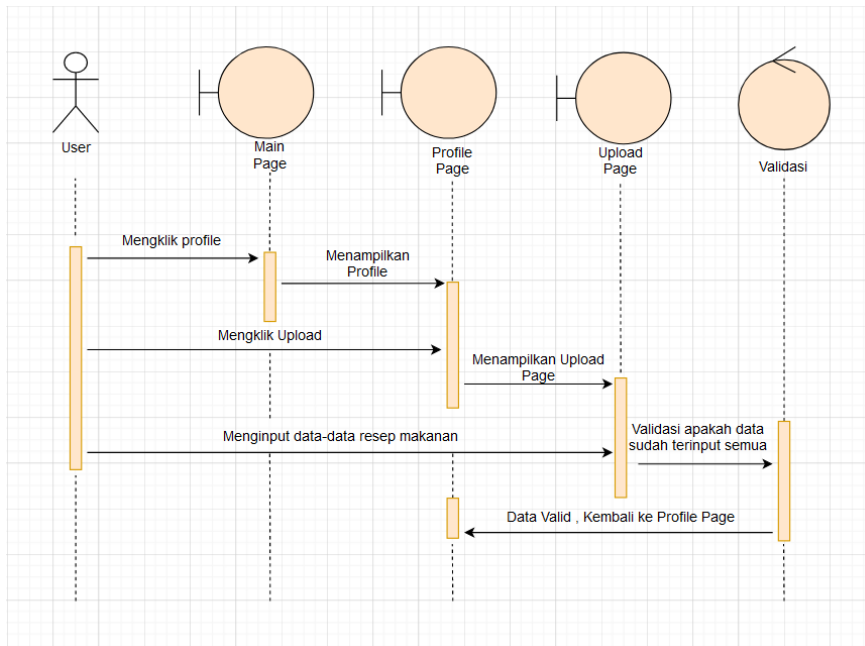
SEARCH BAR



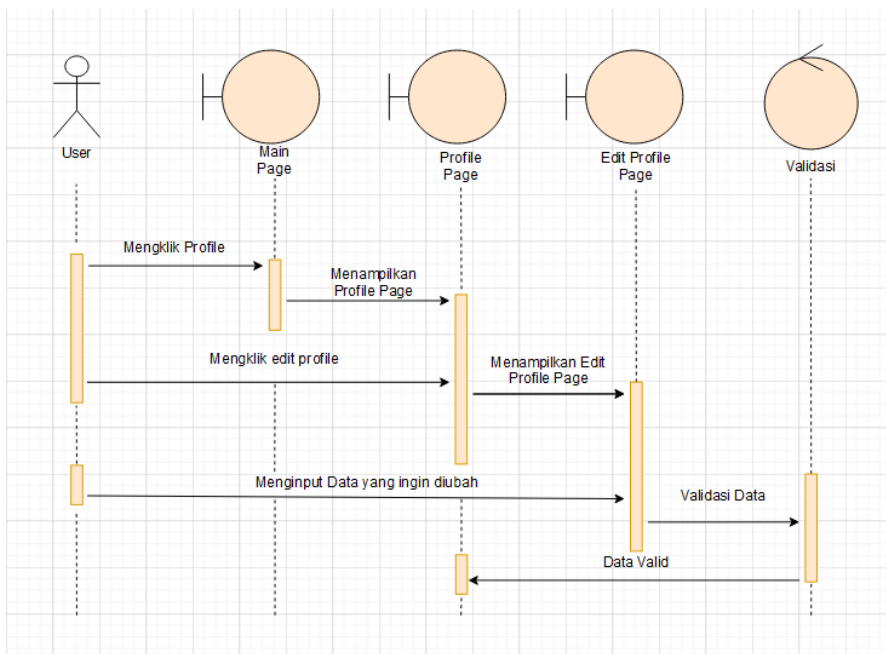
CATEGORY



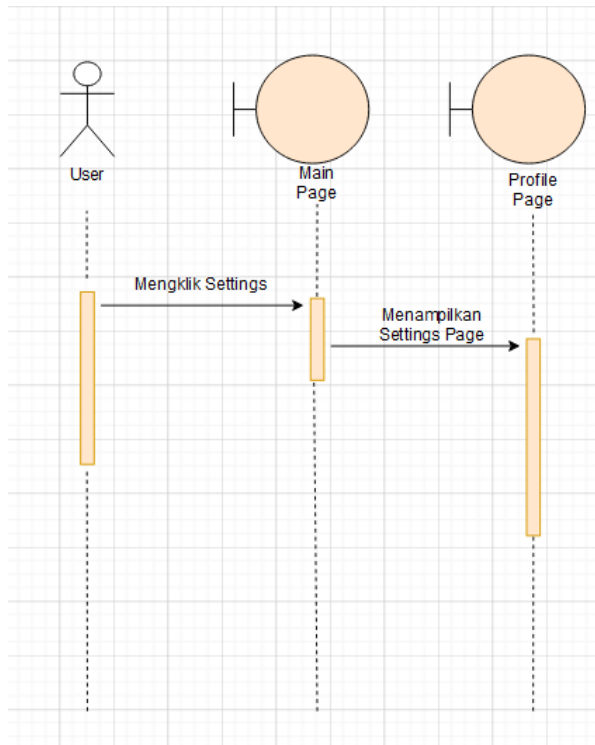
UPLOAD RESEP MAKAN



EDIT PROFILE PAGE

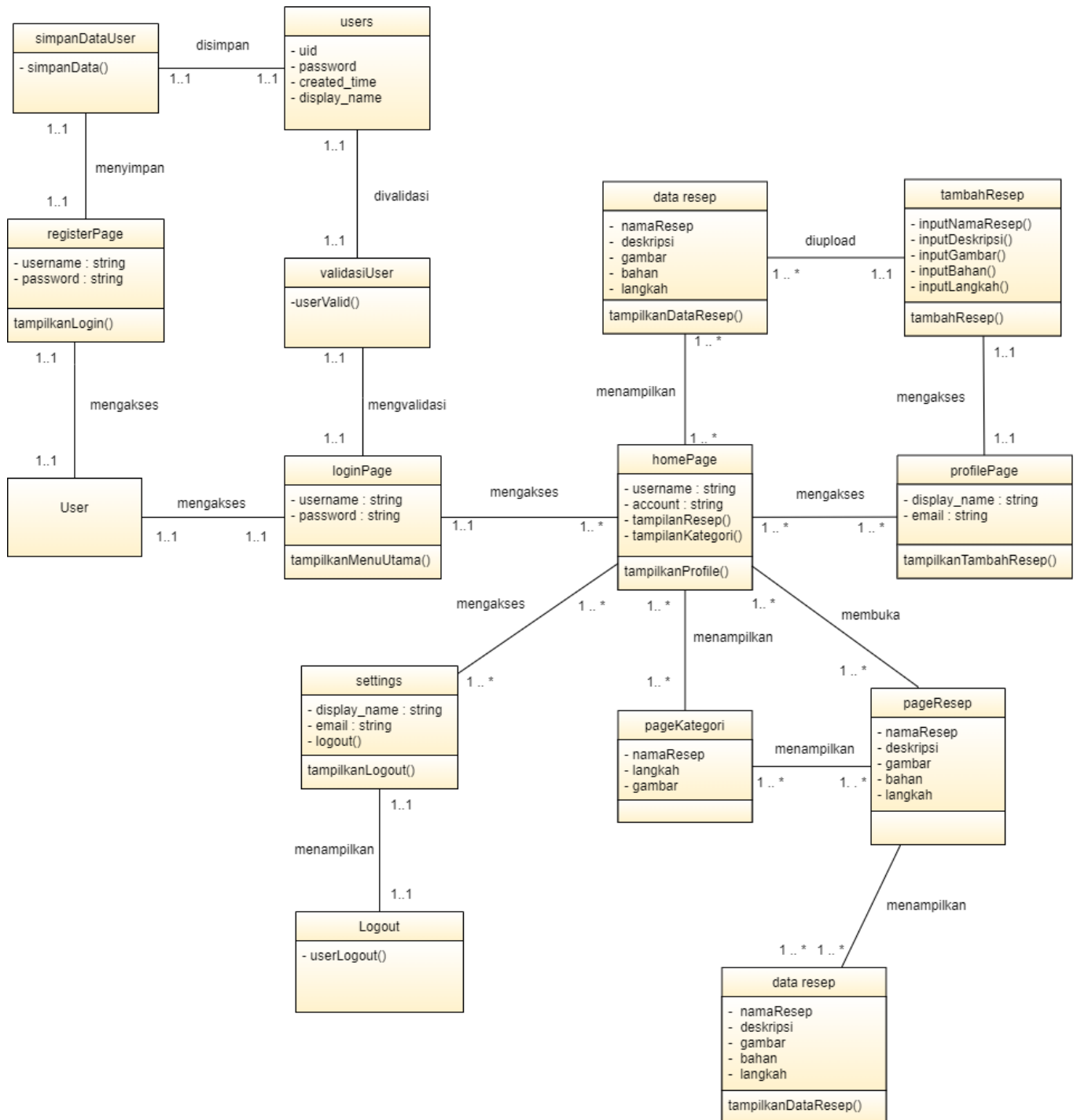


SETTINGS APLIKASI



CLASS DIAGRAM

- class diagram sistem



pada class diagram ini, User pada tahap awal membuka aplikasi dapat mengakses registerPage jika belum memiliki akun. akun yang telah diregister akan disimpan di data user. User juga bisa langsung mengakses loginPage jika sudah memiliki akun. Setelah login, user akan ditampilkan dengan homePage / mainPage yang berisi tampilan

makanan yang jika diklik akan membawa user ke pageResep dimana user dapat melihat resep makanan tersebut secara detail. Pada homePage terdapat profilePage di mana di profilePage user bisa mengupload resep di page tambahResep. Pada page tambahResep user bisa menginput namaResep, deskripsi, gambar, bahan, dan langkah resep makanan. Resep yang disubmit user akan langsung tersimpan di database resep.

- class diagram data



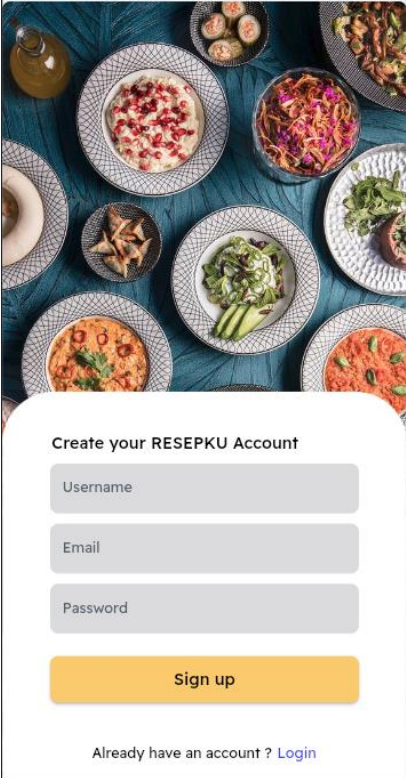
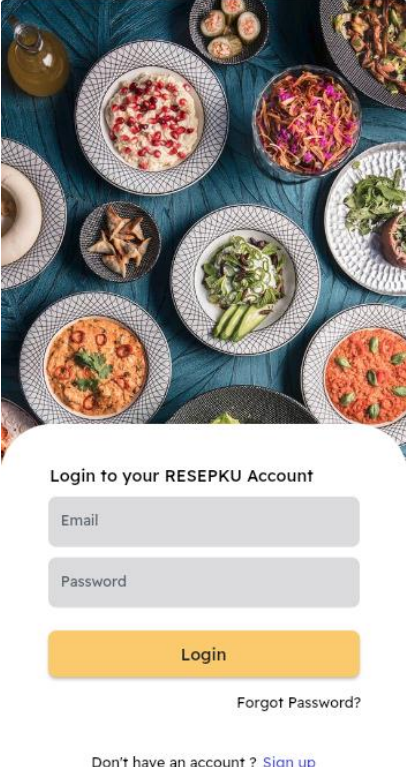
data users adalah data yang disimpan ketika user melakukan registrasi. Data resep adalah tempat data menyimpan resep makanan yang ditambah oleh user, resep tersebut akan terhubung kepada data users melalui uid (user id).



Setiap 2 hari sekali diadakan meeting daily scrum pada jam 9.00 am di tempat meeting kantor pusat.

SPRINT 1 (DURASI 15 HARI)

- Tanggal : 2-20 Oktober 2023 (tidak termasuk sabtu dan minggu)
- Sprint backlog :
 - Sign up dan Sign in aplikasi (**5 poin**)
 - User dapat melihat tampilan mainpage yang berisi tampilan makanan (**8 poin**)
 - User dapat memilih kategori resep makanan (**8 poin**)
- Team velocity : 5+12 : 17 poin
- Poin cerita yang tertunda : 0 poin
- Success Rate : 17/17 : 100%
- Kesimpulan :
Tim telah berhasil menyelesaikan seluruh Sprint backlog dengan kesuksesan 100%, tanpa ada poin cerita yang tertunda. Tugas-tugas utama mencakup pendaftaran dan masuk ke aplikasi, serta pengembangan tampilan utama.

- Proses Increment dan manual setiap page SPRINT 1 :

No	Gambar page	Penjelasan dan manual
1.		<p>Signup Page</p> <p>Tampilan register di mana user harus memasukkan username, email, dan password dan menekan tombol signup. Data user akan disimpan ke dalam database firebase.</p>
2.		<p>Login Page</p> <p>Tampilan login di mana user harus memasukkan email, dan password dan klik tombol login untuk login ke dalam aplikasi.</p>

3.		<p>Home Page</p> <p>page ini merupakan tampilan utama dalam aplikasi dapat disebut juga sebagai home page. di home page terdapat fitur search, kategori, dan resep makanan yang dapat diklik.</p>
4.		<p>Page resep</p> <p>Page ini menampilkan resep makanan dari makanan yang telah diklik user. di dalam page terdapat bahan, langkah memasak dan gambar makanan.</p>


SPRINT 2 (DURASI 15 HARI)

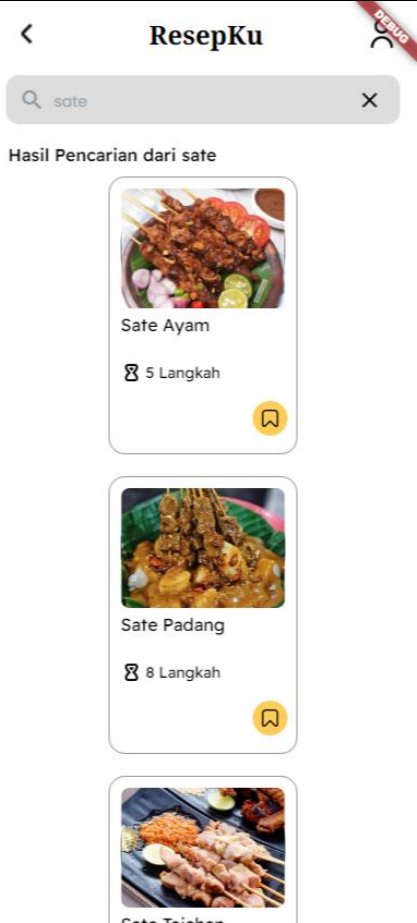
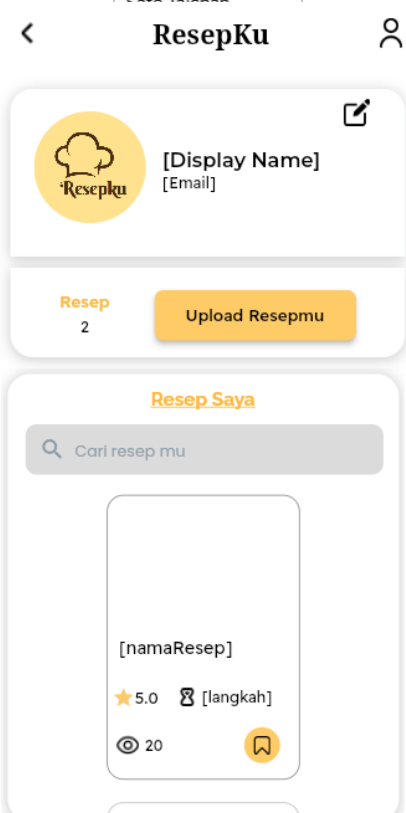
- Tanggal : 23 Oktober 2023 – 10 November 2023 (tidak termasuk sabtu dan minggu)
- Sprint backlog :
 - User dapat memilih category resep makanan (**8 poin**)
 - User dapat mencari resep makanan (**5 poin**)
 - User dapat melihat profile account (**4 poin**)
- Team velocity : 8+5+4 : 17 poin
- Point cerita yang tertunda : 0 poin
- Success Rate : 17/17 : 100%

• Kesimpulan :

Sprint backlog terdiri dari tiga tugas utama yaitu pengembangan kemampuan pengguna untuk memilih kategori resep makanan , melakukan pencarian resep makanan dan melihat profil akun. Tidak ada poin cerita yang tertunda, dan tim berhasil menyelesaikan seluruh Sprint backlog dengan tingkat keberhasilan 100%. Sprint ini telah sukses dalam memenuhi semua target yang telah ditetapkan.

- Proses Increment dan manual dari page SPRINT 2 :

1.		<h3>Kategori Page</h3> <p>Page ini menampilkan kategori makanan sesuai dengan kategori yang dipilih oleh user. Makanan yang ditampilkan diambil dari database resep di firebase.</p>
----	---	--

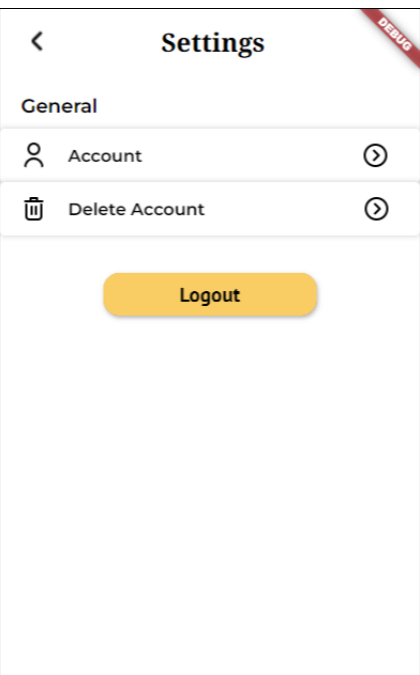
2.		<p>Page search</p> <p>Page ini menampilkan kategori makanan sesuai dengan yang telah disearch oleh user. Makanan yang ditampilkan diambil dari database resep di firebase.</p>
3.		<p>Profile Page</p> <p>Page profile menampilkan profile picture, nama dan email user dari data user firebase. di dalam page profile juga terdapat resep yang telah diupload oleh user.</p>

SPRINT 3 (DURASI 15 HARI)

- Tanggal : 13 November 2023 – 1 Desember 2023 (tidak termasuk sabtu dan minggu)
- Sprint backlog :
 - User dapat mengupload resep (**8 poin**)
 - User dapat membuka settings (**5 poin**)
- Team velocity : 8+5 : 13 point
- Point cerita yang tertunda : 0 poin
- Success Rate : 13/13 : 100%
- Kesimpulan :

Sprint kali ini fokus pada dua aspek utama, yaitu kemampuan pengguna untuk mengunggah resep dan membuka pengaturan pada aplikasi. Dengan tidak ada poin cerita yang tertunda, tim berhasil menyelesaikan seluruh Sprint backlog dengan tingkat keberhasilan 100%. Keberhasilan ini menunjukkan pencapaian yang solid dalam mencapai tujuan Sprint.

- Proses Increment dan manual page SPRINT 3 :

1.		<p>Page settings</p> <p>Page ini menampilkan settings yang terdapat dalam aplikasi. di dalam settings user dapat edit profile, delete account, dan logout.</p>
----	--	---

Upload Resep

Judul Resep

Tulis nama resepmu...

Deskripsi Resep

Tulis deskripsi resepmu...

Lama Memasak

Contoh : 30 Menit

Foto Makanan

Bahan Masakan

Tulis bahan yang diperlukan...

Langkah Memasak Resep

Tulis langkah memasak resep..

Tulis langkah memasak resep..

Tulis langkah memasak resep..

Tambah Langkah

Pilih Kategori Makanan

Nasi

Mie

Ayam

Sate

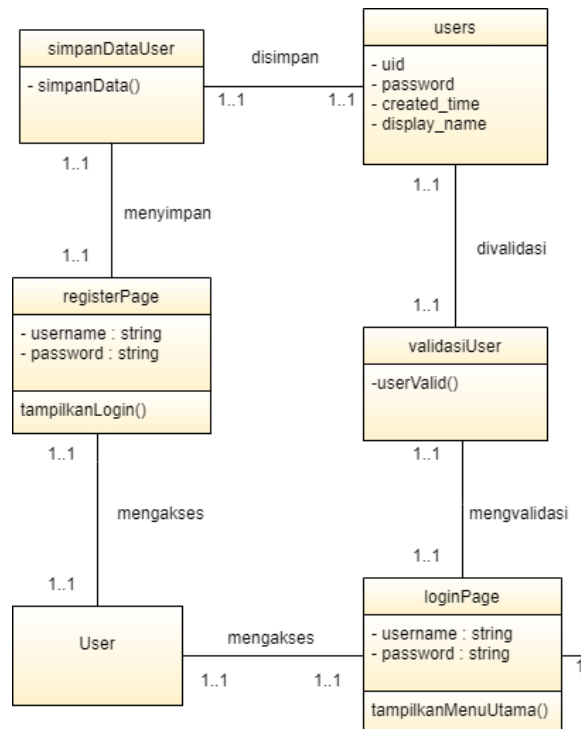
Upload Resep

Page tambah resep

di dalam page ini user dapat mengupload resep mereka. user harus memasukkan judul, deskripsi, foto, bahan makanan dan langkah-langkah memasak resep. Resep yang telah diupload akan disimpan di dalam database resep.

Penjelasan Source Code

1. Register dan loginPage



Register_page_model.dart

```
13 class RegisterPageModel extends FlutterFlowModel<RegisterPageWidget> {
14   final unfocusNode = FocusNode();
15
16   FocusNode? userFieldFocusNode;
17   TextEditingController? userFieldController;
18   String? Function(BuildContext, String?)? userFieldControllerValidator;
19
20   FocusNode? emailFieldFocusNode;
21   TextEditingController? emailFieldController;
22   String? Function(BuildContext, String?)? emailFieldControllerValidator;
23
24   FocusNode? passFieldFocusNode;
25   TextEditingController? passFieldController;
26   late bool passFieldVisibility;
27   String? Function(BuildContext, String?)? passFieldControllerValidator;
28
29 > void initState(BuildContext context) { ...
32
33 > void dispose() { ...
44 }
```

- Line 13 : Membuat kelas RegisterPageModel yang meng-extend FlutterFlowModel dan terkait dengan widget RegisterPageWidget.
- Line 14 : Mendeklarasikan FocusNode bernama unfocusNode untuk menangani unfocus pada field teks.

- Line 16-27 : state field untuk widget input username,email, dan pass termasuk FieldFocusNode untuk penanganan fokus, FieldController untuk input teks, dan FieldControllerValidator untuk validasi.

Register_page_widget.dart

```

14  class RegisterPageWidget extends StatefulWidget {
15    const RegisterPageWidget({Key? key}) : super(key: key);
16
17    @override
18    _RegisterPageWidgetState createState() => _RegisterPageWidgetState();
19  }

```

```

21  class _RegisterPageWidgetState extends State<RegisterPageWidget> {
22    late RegisterPageModel _model;
23
24    final scaffoldKey = GlobalKey<ScaffoldState>();
25
26    @override
27    void initState() {
28      super.initState();
29      _model = createModel(context, () => RegisterPageModel());
30
31      _model.userFieldController ??= TextEditingController();
32      _model.userFieldFocusNode ??= FocusNode();
33
34      _model.emailFieldController ??= TextEditingController();
35      _model.emailFieldFocusNode ??= FocusNode();
36
37      _model.passFieldController ??= TextEditingController();
38      _model.passFieldFocusNode ??= FocusNode();
39
40      WidgetsBinding.instance.addPostFrameCallback((_) => setState(() {}));
41    }

```

```

43    @override
44    void dispose() {
45      _model.dispose();
46
47      super.dispose();
48    }
49
50    @override
51    Widget build(BuildContext context) { ...
52  }

```



```

326         child: FFButtonWidget(
327             onPressed: () async {
328                 GoRouter.of(context).prepareAuthEvent();
329
330                 final user =
331                     await authManager.createAccountWithEmail(
332                         context,
333                         _model.emailFieldController.text,
334                         _model.passFieldController.text,
335                     );

```

- Line 14-19 : membuat kelas RegisterPageWidget sebagai widget stateful.
- Line 21-41 : metode initState untuk inisialisasi state dan controller serta focus node untuk field username, email, dan password.
- Line 44-48 : metode dispose untuk membersihkan sumber daya yang digunakan setelah widget dihapus.
- Line 51-437 : metode build untuk menampilkan tampilan halaman pendaftaran. Termasuk pengaturan tata letak, gambar latar belakang, dan elemen formulir pendaftaran.
- Line 326 – 335 : authManager.createAccountWithEmail adalah metode yang bertanggung jawab untuk membuat akun pengguna baru menggunakan alamat email dan kata sandi yang di ambil dari _model.emailFieldController.text dan pass.FieldController.text Metode ini melibatkan interaksi dengan Firebase Authentication.

Login_page_model.dart

```

11 class LoginPageModel extends FlutterFlowModel<LoginPageWidget> {
12     final unfocusNode = FocusNode();
13
14     FocusNode? emailsuerFocusNode;
15     TextEditingController? emailsuerController;
16     String? Function(BuildContext, String?)? emailsuerControllerValidator;
17
18     FocusNode? passuserFocusNode;
19     TextEditingController? passuserController;
20     late bool passuserVisibility;
21     String? Function(BuildContext, String?)? passuserControllerValidator;
22
23     void initState(BuildContext context) {
24         passuserVisibility = false;
25     }
26
27 > void dispose() { ...
35
36

```

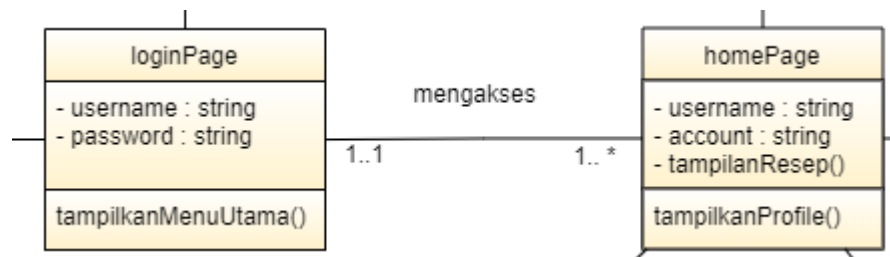
Line 11-12 : membuat kelas LoginPageModel dengan extend FlutterFlowModel dengan parameter tipe LoginPageWidget. Membuat objek FocusNode bernama unfocusNode yang dapat digunakan untuk menghilangkan fokus pada elemen input.

Line 14-21 : Mendeklarasikan field terkait dengan elemen input email dan pass pada halaman login, seperti `userFocusNode` (untuk fokus), `userController` (untuk mengontrol input), dan `userControllerValidator` (validator untuk validasi).

Line 23 – 25 : metode `initState` untuk melakukan inisialisasi state pada saat widget halaman `passuserVisibility` menjadi `false`.

Line 27 – 35 : `void dispose` untuk membersihkan sumber daya yang digunakan oleh objek `FocusNode` dan `TextEditingController` setelah widget dihapus atau tidak diperlukan lagi.

2. loginPage mengakses ke homePage



Login_page_widget.dart

```
12 > class LoginPageWidget extends StatefulWidget { ...
18
19 class _LoginPageWidgetState extends State<LoginPageWidget> {
20   late LoginPageModel _model;
21
22   final scaffoldKey = GlobalKey<ScaffoldState>();
23
24   @override
25   void initState() {
26     super.initState();
27     _model = createModel(context, () => LoginPageModel());
28
29     _model.emailsuerController ??= TextEditingController();
30     _model.emailsuerFocusNode ??= FocusNode();
31
32     _model.passuserController ??= TextEditingController();
33     _model.passuserFocusNode ??= FocusNode();
34
```

```

258   child: FFButtonWidget(
259     onPressed: () async {
260       GoRouter.of(context).prepareAuthEvent();
261
262       final user = await authManager.signInWithEmail(
263         context,
264         _model.emailsuerController.text,
265         _model.passuserController.text,
266       );
267       if (user == null) {
268         return;
269       }
270
271       context.goNamedAuth(
272         'homePageLama', context.mounted);
273     },

```

Line 12 : membuat class LoginPageWidget sebagai widget stateful

Line 29 -33 : Inisialisasi kontroler dan node fokus untuk widget input email dan password.

Line 258 – 273 : Pada saat tombol ditekan, menggunakan authManager.signInWithEmail untuk melakukan otentikasi pengguna menggunakan email dan password yang dimasukkan. Jika otentikasi berhasil, pengguna dialihkan ke halaman 'homePageLama'.

```

15 class HomePageWidget extends StatefulWidget {
16   const HomePageWidget({Key? key}) : super(key: key);
17
18   @override
19   _HomePageWidgetState createState() => _HomePageWidgetState();
20 }
21
22 class _HomePageWidgetState extends State<HomePageWidget> {
23   late HomePageModel _model;
24
25   final scaffoldKey = GlobalKey<ScaffoldState>();
26
27   @override
28   void initState() {
29     super.initState();
30     _model = createModel(context, () => HomePageModel());
31
32     _model.textController ??= TextEditingController();
33     _model.textFieldFocusNode ??= FocusNode();
34
35     WidgetsBinding.instance.addPostFrameCallback((_) => setState(() {}));
36   }
37
38   @override
39   void dispose() {
40     _model.dispose();
41
42     super.dispose();
43   }
44
45   @override
46   Widget build(BuildContext context) { ...
1494 }

```

Line 15-20 : membuat kelas HomePageWidget yang merupakan widget Stateful.

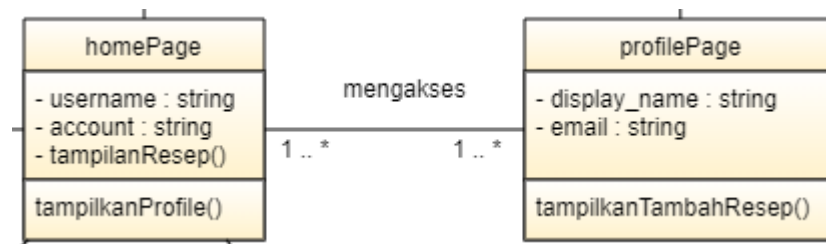
Line 22 : membuat kelas _HomePageWidgetState, yang merupakan state dari HomePageWidget.

Line 28-30 : Menginisialisasi instance dari HomePageModel menggunakan fungsi createModel.

Line 32-33 : Menginisialisasi kontroler teks dan node fokus untuk widget teks.

Line 46 : Mendeklarasikan metode build yang akan mengembalikan widget untuk merender halaman homePage.

3. homePage mengakses ke profilePage



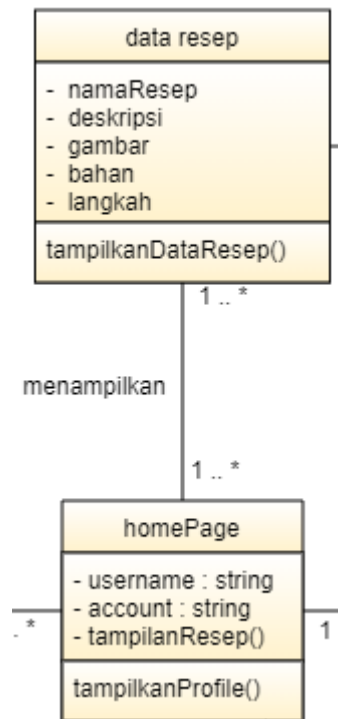
home_page_widget.dart // page yang menampilkan makanan, profile, dan settings

```
139 child: InkWell(
140     splashColor: Colors.transparent,
141     focusColor: Colors.transparent,
142     hoverColor: Colors.transparent,
143     highlightColor: Colors.transparent,
144     onTap: () async {
145         context.pushNamed('profilePage');
146     },
147     child: ClipRRect(
148         borderRadius: BorderRadius.circular(8.0),
149         child: Image.asset(
150             'assets/images/user(2).png',
151             width: 27.0,
152             height: 28.0,
153             fit: BoxFit.contain,
154             alignment: Alignment(0.00, 0.00),
155         ), // Image.asset
156     ), // ClipRRect
157 ), // InkWell
```

Line 144-146 : ketika widget di tap, akan mengarahkan pengguna ke halaman dengan nama 'profilePage' menggunakan context.pushNamed('profilePage').

Line 147 – 154 : widget ClipRRect digunakan untuk memotong sudut gambar dengan radius tertentu dan widget Image.asset yang menampilkan gambar dari berkas di dalam proyek.

4. homePage menampilkan makanan dari dataresep



Home_page_widget.dart // page yang menampilkan makanan, profile, dan settings

```
611 child: Row(  
612   mainAxisAlignment: MainAxisAlignment.center,  
613   children: [  
614     Align(  
615       alignment: AlignmentDirectional(0.00, 0.00),  
616       child: Padding(  
617         padding: EdgeInsetsDirectional.fromSTEB( // EdgeInsetsDirectional  
618       ),  
619       child: InkWell(  
620         splashColor: Colors.transparent,  
621         focusColor: Colors.transparent,  
622         hoverColor: Colors.transparent,  
623         highlightColor: Colors.transparent,  
624         onTap: () async {  
625           context.pushNamed('resepNasgor');  
626         },  
627         child: Container(  
628           width:  
629             MediaQuery.sizeOf(context).width * 0.44,  
630           height: 255.0,  
631           decoration: BoxDecoration( // BoxDecoration ...  
632
```

Line 611-632 : sebuah widget interaktif yang menampilkan gambar dan teks "Nasi Goreng". Ketika widget tersebut ditekan, pengguna akan diarahkan ke halaman 'resepNasgor'. Widget ini menggunakan `Padding`, `Row`, dan `Align` untuk mengatur tata letak, serta `InkWell` untuk menanggapi sentuhan pengguna. `Container` digunakan untuk menentukan ukuran dan dekorasi dari widget tersebut.

```

680 child: ClipRRect(
681   borderRadius:
682     BorderRadius.circular(
683       8.0), // BorderRadius.circular
684   child: Image.asset(
685     'assets/images/Nasi_Goreng_Gila_Super_Lezat_-_Resep_ResepKoki.jpg',
686     width: 300.0,
687     height: 200.0,
688     fit: BoxFit.cover,

```

Line 680-688 : widget child ini digunakan untuk menampilkan gambar dengan menggunakan Image.asset

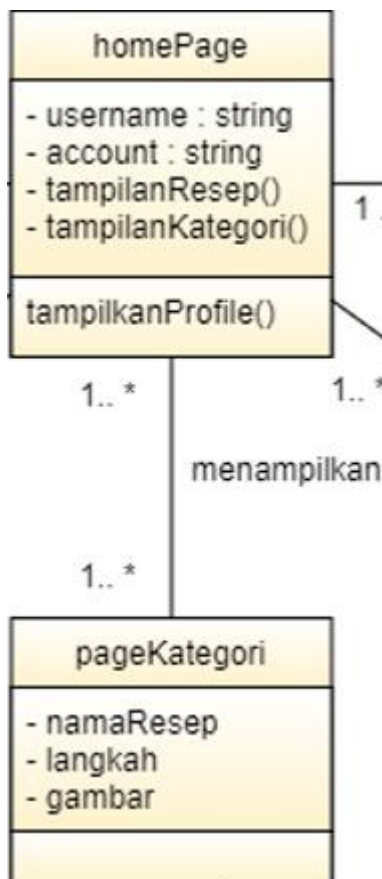
```

702 child: Text(
703   'Nasi Goreng Gila',
704   style:
705     FlutterFlowTheme.of(context)
706       .labellarge
707       .override(
708         fontFamily:
709           'Readex Pro',
710         color: Colors.black,
711         fontSize: 16.0,
712       ),
713 ), // Text

```

Line 702-713 : child text ini bertanggung jawab untuk menampilkan teks 'Nasi Goreng Gila' di bawah gambar. Padding digunakan untuk memberikan jarak antara tepi gambar dan teks, kemudian Text digunakan untuk menampilkan teks tersebut.

5. homePage menampilkan pageKategori



Home_page_widget.dart // page yang menampilkan makanan, profile, dan settings

```
410 child: InkWell(  
411   splashColor: Colors.transparent,  
412   focusColor: Colors.transparent,  
413   hoverColor: Colors.transparent,  
414   highlightColor: Colors.transparent,  
415   onTap: () async {  
416     context.pushNamed('katNasi');  
417   },  
418   child: ClipRRect(  
419     borderRadius:  
420       BorderRadius.circular(50.0),  
421     child: Image.asset(  
422       'assets/images/Make_Kimchi_Bokumbap,_Korean_Kimchi_Fried_Rice_With_Beef.',  
423       width: 300.0,  
424       height: 200.0,  
425       fit: BoxFit.fill,  
426     ), // Image.asset  
427   ), // ClipRRect  
428 ), // InkWell  
429 ), // Container  
430 padding:  
431   EdgeInsetsDirectional.fromSTEB(  
432     0.0, 5.0, 0.0, 0.0), // EdgeInsetsDirectional.fromSTEB  
433   child: Text(  
434     'Nasi',  
435     style: FlutterFlowTheme.of(context)  
436       .bodyMedium  
437       .override(  
438         fontFamily: 'Readex Pro',  
439         color: Colors.black,  
440       ),  
441   ), // Text  
442 ), // Padding  
443 ],
```

Line 410-415 : Menggunakan InkWell untuk membuat item dapat diklik dan ketika diklik, user akan menuju ke page 'katNasi' yang merupakan resep-resep berkategori nasi

Line 418-427 : Menggunakan ClipRRect untuk membuat sudut gambar menjadi bulat dengan radius 50.0, Menggunakan Image.asset untuk menampilkan gambar dari asset.

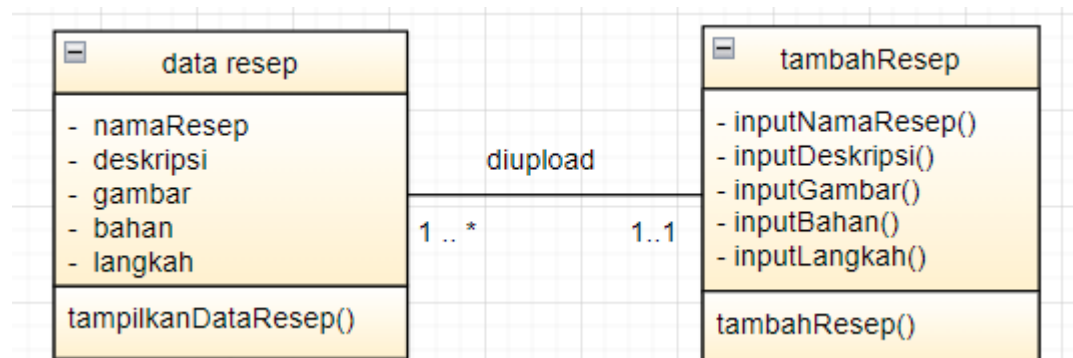
Line 433-441 : Menampilkan teks 'Nasi' sebagai salah satu kategori resep.

Kategori_sate_widget.dart // page yang menampilkan makanan kategori sate

```
2  StreamBuilder<List<ResepRecord>>(  
3    stream: queryResepRecord(  
4      queryBuilder: (resepRecord) => resepRecord.where(  
5        'kategori',  
6        isEqualTo: 'Sate',  
7      ),  
8    ),
```

Line 2-6 : Menerima stream dari fungsi queryResepRecord yang mengembalikan daftar resep yang difilter berdasarkan kategori 'Sate'.

6. menambah resep pada data resep



Upload_resep_page.dart // page untuk menambah resep

```
274      child: TextFormField(  
275        controller: _model.textController2,  
276        focusNode: _model.textFieldFocusNode2,  
277        autofocus: true,  
278        obscureText: false,  
279        decoration: InputDecoration(  
280          labelText: 'Tulis deskripsi resepmu...',  
281          labelStyle: FlutterFlowTheme.of(context)  
282            .labelMedium  
283            .override(  
284              fontFamily: 'Readex Pro',  
285              color: Colors.black,  
286            ),  
287        ),
```

Line 274-286 : child yang berisi TextFormField untuk menginput deskripsi resep

```

452 child: InkWell(
453   splashColor: Colors.transparent,
454   focusColor: Colors.transparent,
455   hoverColor: Colors.transparent,
456   highlightColor: Colors.transparent,
457   onTap: () async {
458     final selectedMedia =
459       await selectMediaWithSourceBottomSheet(
460         context: context,
461         allowPhoto: true,
462       );
463     if (selectedMedia != null &&
464       selectedMedia.every((m) =>
465         validateFileFormat(
466           m.storagePath, context))) {
467       setState(() =>
468         _model.isDataUploading1 = true);
469       var selectedUploadedFiles =
470         <FFUploadedFile>[];
471
472       var downloadUrls = <String>[];
473       try {
474         selectedUploadedFiles = selectedMedia
475           .map((m) => FFUploadedFile(
476             name: m.storagePath
477               .split('/')
478               .last,
479             bytes: m.bytes,
480             height:
481               m.dimensions?.height,
482             width: m.dimensions?.width,
483             blurHash: m.blurHash,
484           )) // FFUploadedFile
485           .toList();
486
487       downloadUrls = (await Future.wait(
488         selectedMedia.map(
489           (m) async => await uploadData(
490             m.storagePath, m.bytes),

```

Line 452-462 : Ketika area ini diklik (onTap), Menggunakan fungsi 'selectMediaWithSourceBottomSheet' untuk memunculkan bottom sheet yang memungkinkan user memilih media foto sesuai dengan fungsi 'allowPhoto : true'.

Line 463-466 : Memeriksa apakah user telah memilih media (selectedMedia != null) dan apakah media yang dipilih sesuai dengan format file yang diizinkan. Validasi ini menggunakan fungsi validateFileFormat.

Line 472-490 : Membuat list dari URL download media yang diunggah dengan melakukan upload data menggunakan fungsi uploadData.

```

1060 child: FFButtonWidget(
1061   onPressed: () async {
1062     await ResepRecord.collection.doc().set({
1063       ...createResepRecordData(
1064         namaResep: _model.textController1.text,
1065         deskripsi: _model.textController2.text,
1066         langkah: _model.textController3.text,
1067         gambar: _model.uploadedFileUr11,
1068         uid: currentUserUid,
1069         kategori: _model.radioButtonValue,
1070       ),
1071       ...mapToFirestore(
1072         {
1073           'bahanMasak':
1074             _model.bahanmasakModels.getValues(
1075               (m) => m.textController.text,
1076             ),
1077           'langkahMasak':
1078             _model.formcobaModels.getValues(
1079               (m) => m.textController.text,
1080             ),
1081         },
1082       ),
1083     });
1084
1085     context.pushNamed('profilePage');
1086   },
1087   text: 'Upload Resep',
1088   options: FFButtonOptions(
1089     width: 300.0,
1090     height: 40.0,
1091     padding: EdgeInsetsDirectional.fromSTEB(

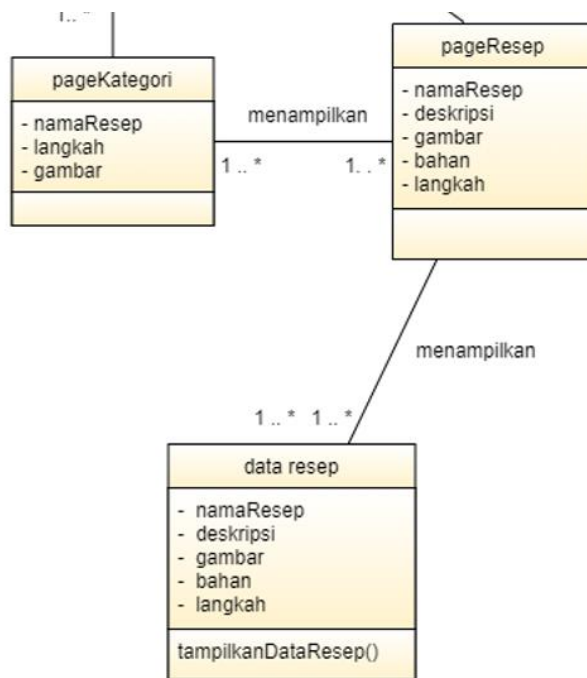
```

Line 1060-1084 : Ketika tombol diklik, data resep baru diunggah ke koleksi ResepRecord di Firestore menggunakan set. Data resep melibatkan informasi seperti nama resep, deskripsi, langkah-langkah, gambar, uid pengguna, dan kategori.

Line 1085 : setelah data resep diupload, user dialihkan ke halaman profilePage menggunakan context.pushNamed.

Line 1087-1091 : widget text yang berisi 'Upload Resep'.

7. user membuka resepMakanan



Kategori_page.dart // page sesuai kategori makanan tertentu

```

    onTap: () async {
      await Share.share(
        columnResepRecord
          .namaResep,
        sharePositionOrigin:
          getWidgetBoundingBox(
            context),
      );

      context.pushNamed(
        'resepMakan',
        queryParameters: {
          'namaResep':
            serializeParam(
              columnResepRecord
                .namaResep,
              ParamType
                .String,
            ),
          'langkah':
            serializeParam(
              columnResepRecord
                .langkah,
              ParamType
                .String,
            ),
          'gambar':
            serializeParam(
              columnResepRecord
                .gambar,
              ParamType
                .String,
            ),
        }.withoutNulls,
      );
    },
  ),

```

Penjelasan : context.pushNamed 'resepMakan' digunakan untuk melakukan navigasi ke halaman 'resepMakan'. parameter juga disertakan menggunakan queryParameters. Parameter tersebut adalah informasi resep yang akan ditampilkan di halaman resepMakan.

Resep_makan_page.dart // page untuk menampilkan resep makanan

```
12 class ResepMakanWidget extends StatefulWidget {
13   const ResepMakanWidget({
14     Key? key,
15     required this.namaResep,
16     required this.langkah,
17     required this.gambar,
18   }) : super(key: key);
19
20   final String? namaResep;
21   final String? langkah;
22   final String? gambar;
23
24   @override
25   ResepMakanWidgetState createState() => _ResepMakanWidgetState();
26 }
27
```

Line 12 : membuat kelas ResepMakanWidget dengan widget stateful

Line 13-22 : ResepMakanWidget menerima tiga parameter wajib (namaResep, langkah, dan gambar). Ketiga parameter tersebut adalah informasi yang dibutuhkan untuk menampilkan resep makanan dalam widget ini.

8. User search makanan

Search_page_widget.dart

```
2  if (!_model.searchActive)
3    Builder(
4      builder: (context) {
5        final searchMakan = searchResepRecordList.toList();
6        return ListView.builder(
7          padding: EdgeInsets.zero,
8          primary: false,
9          shrinkWrap: true,
10         scrollDirection: Axis.vertical,
11         itemCount: searchMakan.length,
12         itemBuilder: (context, searchMakanIndex) {
13           final searchMakanItem = searchMakan[searchMakanIndex];
14           return Padding(
```

Line 2 : menggunakan If untuk cek apakah pageState searchActive bernilai false. Jika bernilai false maka akan menampilkan semua resep, dan jika user melakukan searching pageState searchActive akan menjadi true.

```

211         child: Padding(
212           padding:
213             EdgeInsetsDirectional.fromSTEB(
214               4, 0, 0, 0),
215           child: TextFormField(
216             controller: _model.textController,
217             focusNode:
218               _model.textFieldFocusNode,
219             onFieldSubmitted: (_) async {
220               safeSetState(() {
221                 _model.simpleSearchResults =
222                   TextSearch(
223                     searchResepRecordList
224                       .map(
225                         (record) =>
226                           TextSearchItem
227                             .fromTerms(
228                               record, [
229                                 record.namaResep!,
230                                 record.kategori!
231                               ]),
232                         )
233                       .toList(),
234                     )
235                       .search(_model
236                         .textController
237                           .text)
238                       .map((r) => r.object)
239                       .toList();
240               });
241             });
242           setState(() {
243             _model.searchActive = true;
244           });
245         },
246         autofocus: true,
247         obscureText: false,
248         decoration: InputDecoration(
249           hintText:
250             'Cari resep makanan di sini..',

```

Line 215 = widget TextFormField yang menyediakan input teks yang dapat diedit oleh pengguna.

Line 219-245 = ketika user menekan tombol "submit" di keyboard akan dilakukan pemrosesan pencarian dengan menggunakan TextSearch yang akan mencari resep makanan dari firebase sesuai dengan yang diinput oleh user.

Line 242-243 = pageState searchActive value diset menjadi true dan akan menampilkan listview khusus resep makanan sesuai yang disearch oleh user.

