Test Application Design

PT. XYZ adalah sebuah perusahaan fintech yang ingin mengembangkan mobile apps mereka, dalam upaya menjangkau pengguna yang lebih luas mereka ingin mengembakan aplikasi pinjaman online.

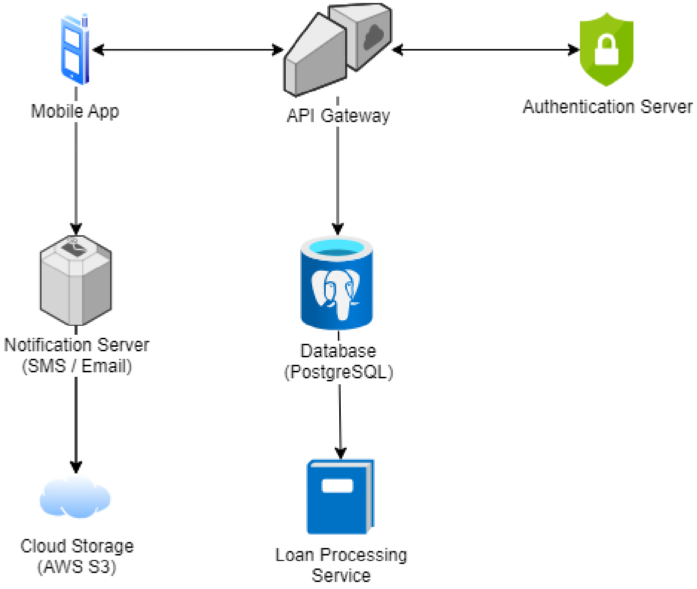
Potential High Level User Story:

- User melakukan registrasi dengan data dir,emaili, nomor telepon dan upload foto beserta KTP
- User dapat login dengan password atau biometric (jika ada di perangkat mobilenya)
- User dapat melihat Sisa hutang dan tagihan perbulan yang harus di bayarkan (Jika ada)
- User dapat meminjam uang paling besar Rp. 12.000.000 dengan tenor maksimal 1 taun.
- User dalam proses peminjaman akan di proses dengan hasil diterima atau ditolak
- Jika pinjaman diterima maka akan ada notifikasi lewat email dan nomot telepon yang terdaftar
- User tidak dapat melakukan peminjaman uang jika sedang ada proses peminjaman dan belum di lunaskan.

Tugas Anda:

1. Buatlah high level design architecture atas project mobile apps ini.

Components High Level Design Architecture:



a. **Mobile Application:**
   i. Platform: iOS and Android.
   ii. Language: Swift (iOS) and Kotlin (Android) or Flutter (Dart)

b. Backend Server:
   i. Server: Node.js with Express.js framework.

ii. Database: PostgreSQL.
iii. Authentication: JWT for token-based authentication.
iv. Cloud Storage: AWS S3 for storing user photos and KTP images.

c. **Third-Party Integrations:**

i. SMS and Email Service: Twilio or SendGrid for notifications.
ii. Biometric Authentication: Integrated with device capabilities.

d. **Security:**

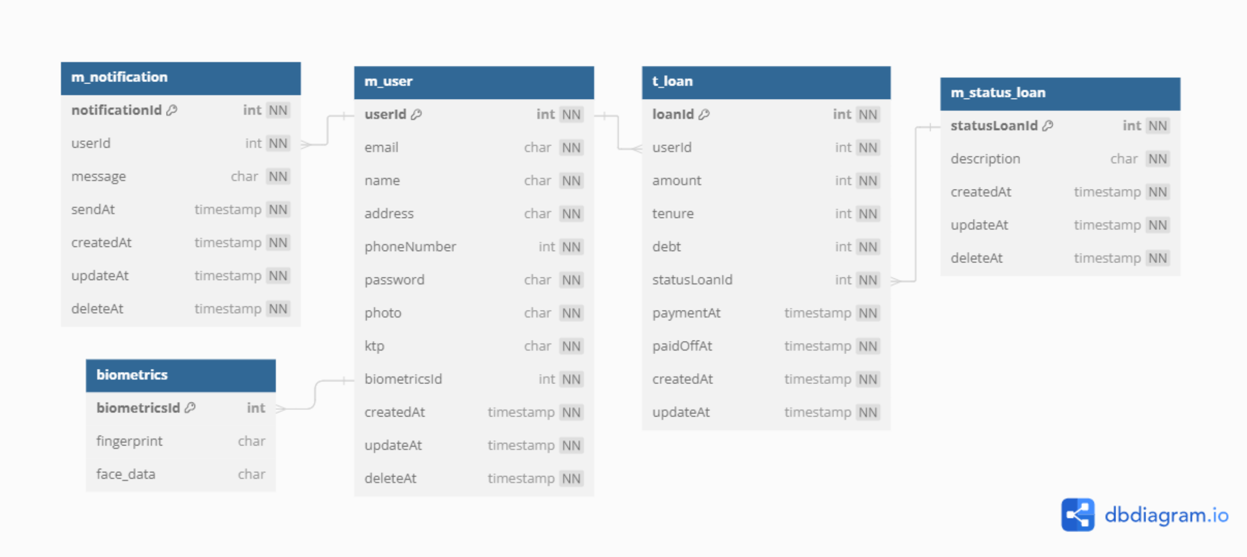
i. HTTPS for secure communication.
ii. Encryption for sensitive data (e.g., KTP images, passwords).

2. Spesifikasikan design Screen Flow dan ERD atas rancangan yang ingin anda buat.

Screen Flow:

1. Welcome Screen: Options to Register or Login.
2. Registration Screen: Form for data input (email, phone number, photo, KTP upload).
3. Login Screen: Options for password or biometric login.
4. Dashboard: Displays outstanding loans and monthly bills.
5. Loan Application Screen: Form to apply for a loan (amount, tenure).
6. Loan Status Screen: Displays status of loan applications (approved, rejected, pending, paid, outstanding bills).
7. Notifications: Display email and SMS notifications for loan approval.

ERD (Entity-Relationship Diagram):



m_notification		
notificationId	**int**	NN
userId	int	NN
message	char	NN
sendAt	timestamp	NN
createdAt	timestamp	NN
updateAt	timestamp	NN
deleteAt	timestamp	NN

m_user		
userId	**int**	NN
email	char	NN
name	char	NN
address	char	NN
phoneNumber	int	NN
password	char	NN
photo	char	NN
ktp	char	NN
biometricsId	int	NN
createdAt	timestamp	NN
updateAt	timestamp	NN
deleteAt	timestamp	NN

t_loan		
loanId	**int**	NN
userId	int	NN
amount	int	NN
tenure	int	NN
debt	int	NN
statusLoanId	int	NN
paymentAt	timestamp	NN
paidOffAt	timestamp	NN
createdAt	timestamp	NN
updateAt	timestamp	NN

m_status_loan		
statusLoanId	**int**	NN
description	char	NN
createdAt	timestamp	NN
updateAt	timestamp	NN
deleteAt	timestamp	NN

biometrics	
biometricsId	**int**
fingerprint	char
face_data	char

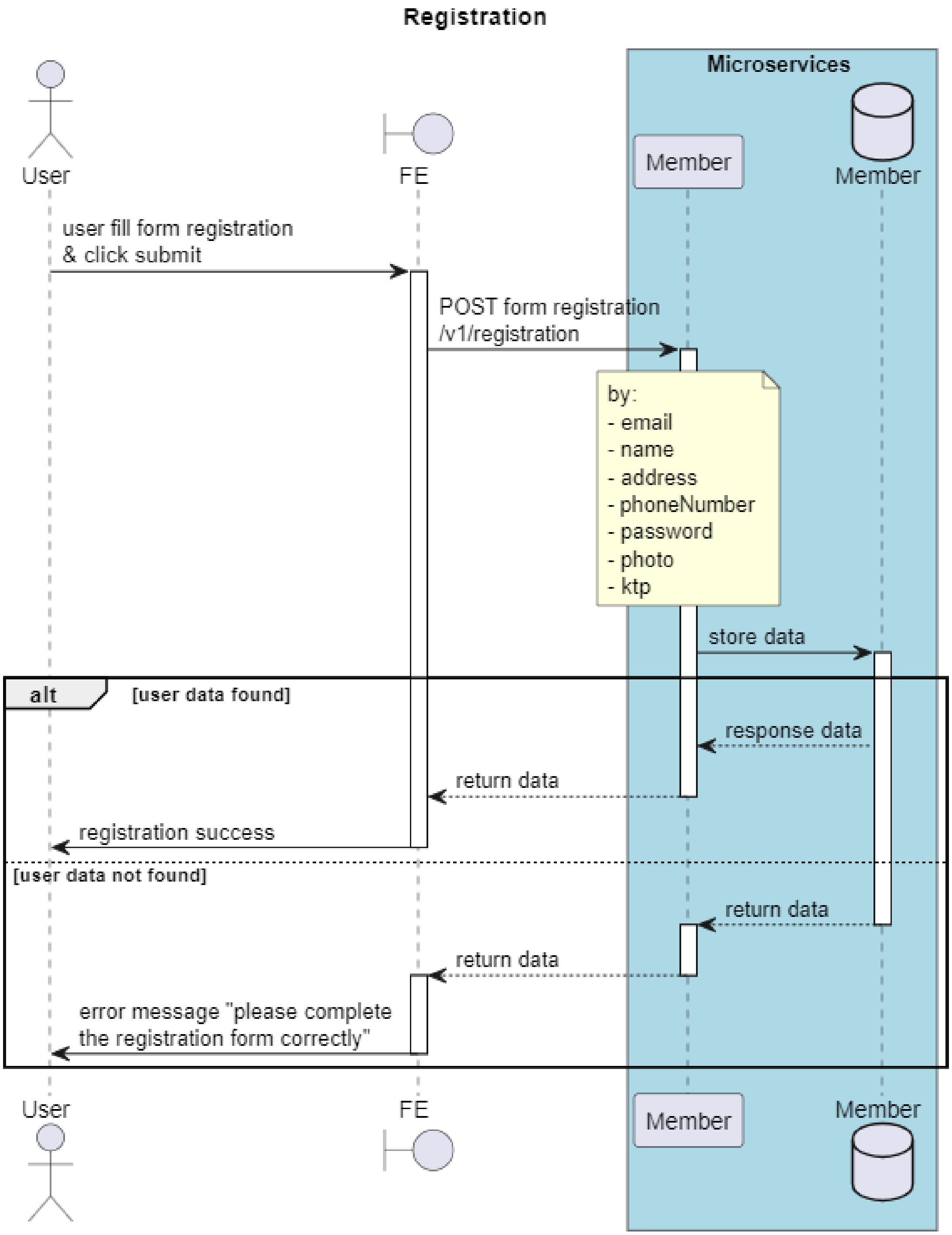
Note :

- Status loan description : approved, rejected, pending

3. Buatlah detail design untuk API dengan menggunakan tools design seperti UML, ERD,flowchart etc.

User Registration

Sequence Diagram



API Contract

Method	POST
Endpoint	{{host}}ms-members/v1/registration
Request	Header

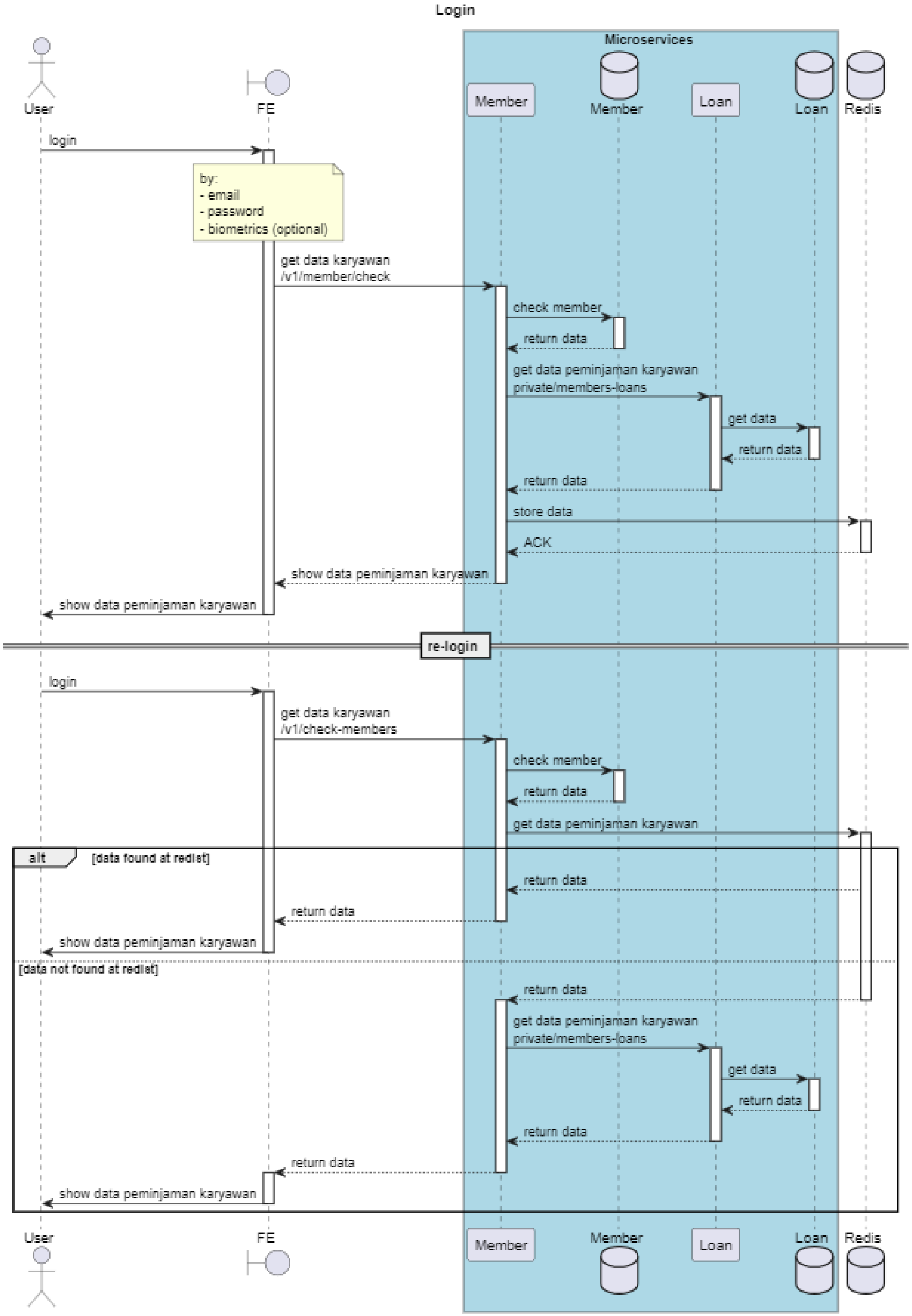
	X-localization: {{LANGUAGE}}
	Content-Type:application/json
	x-api-Key: {{API_KEY}}
	channel: {{CHANNEL}}

	Body
	{
	"name": "sukijan",
	"email": "user@example.com",
	"phoneNumber": "08123456789",
	"password": "password123",
	"photo": "base64-encoded-image",
	"ktp": "base64-encoded-image"
	}

Response	**success**
	{
	"status": true,
	"code": "RU001",
	"message": true,
	"errorMessage": null,
	"errorData": null,
	"data": {
	"message": "Registration successful, please check email to verify",
	"member": {
	"memberId": "1234",
	"name": "test",
	"email": "coba@gmail.com"
	}
	}
	}
	failed
	{
	"status": true,
	"code": "RU002",
	"message": null,
	"errorMessage": "Registration failed",
	"errorData": {
	"errorTitle": "Registration Failed",
	"errorDesc": "Please check your email/password"
	},
	"data": null
	}

User Auth Login

Sequence Diagram



API Contract

Method	POST
Endpoint	{{host}}ms-members/v1/check-member

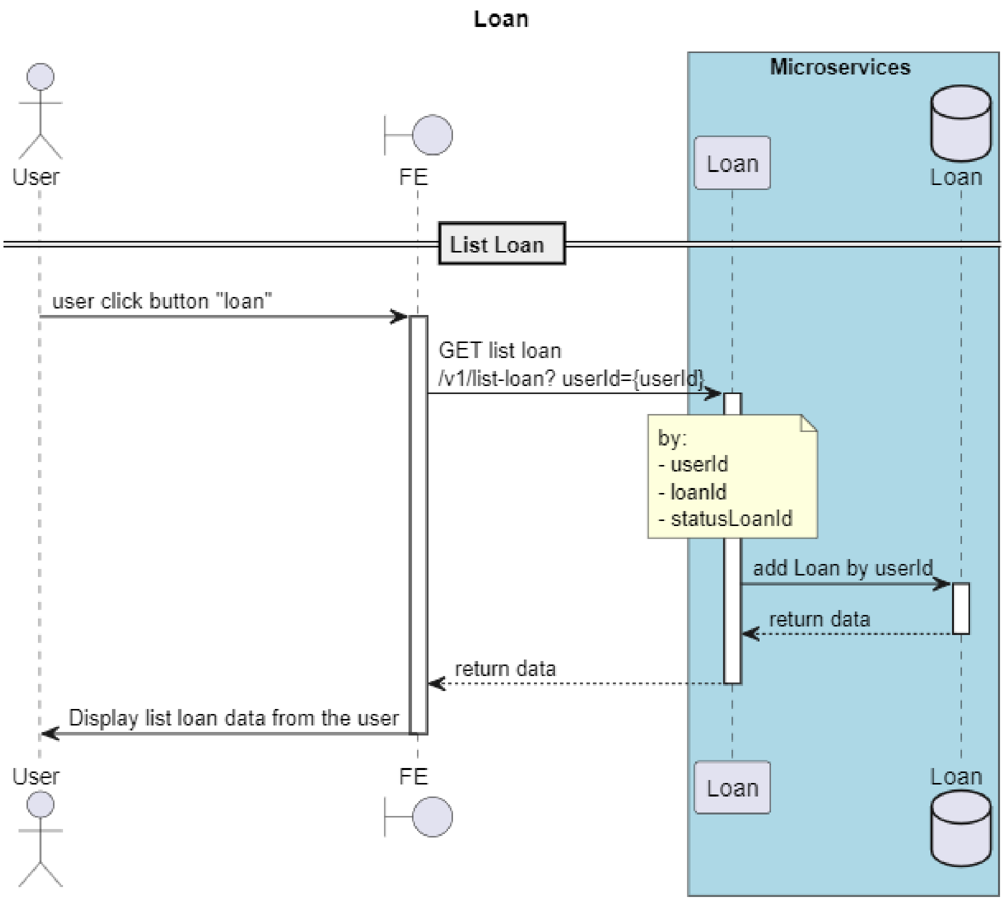
<table>
<tr><td>Request</td><td>Header
X-localization: {{LANGUAGE}}
Content-Type:application/json
x-api-Key: {{API_KEY}}
channel: {{CHANNEL}}</td></tr>
<tr><td></td><td>Body
{
"email": "user@example.com",
"password": "password123",
"biometric" : " biometric123"
}</td></tr>
<tr><td>Response</td><td>**success**
{
"status": true,
"code": "LU001",
"message": true,
"errorMessage": null,
"errorData": null,
"data": {
"message": "Login successful",
"token": "jwt-token",
"member": {
"memberId": "1234",
"name": "test",
"email": "coba@gmail.com",
"phoneNumber": "08123456789",
"photo": "base64-encoded-image",
"ktp": "base64-encoded-image"
},
"loans": [
{
"loanId": "1",
"amount": 5000000,
"debt": 5000000,
"tenure": 12,
"status": {
"statusId" : "1",
"description" : "approved" || "diterima"
},
"createdAt": "2023-06-01T00:00:00Z",
"updateAt": "2023-06-01T00:00:00Z"
},
{
"loanId": "2",</td></tr>
</table>

```
      "amount": 15000000,
      "debt": 15000000,
      "tenure": 12,
      "status": {
        "statusId" : "2",
        "description" : "rejected" || "ditolak"
      },
      "createdAt": "2023-06-01T00:00:00Z",
      "updateAt": "2023-06-01T00:00:00Z"
    },
    {
      "loanId": "3",
      "amount": 5000000,
      "debt": 5000000,
      "tenure": 12,
      "status": {
        "statusId" : "3",
        "description" : "pending" || "menunggu"
      },
      "createdAt": "2023-06-01T00:00:00Z",
      "updateAt": "2023-06-01T00:00:00Z"
    },
    {
      "loanId": "4",
      "amount": 5000000,
      "debt": 0,
      "tenure": 12,
      "status": {
        "statusId" : "4",
        "description" : "paid off" || "lunas"
      },
      "createdAt": "2023-06-01T00:00:00Z",
      "updateAt": "2023-06-01T00:00:00Z"
    },
    {
      "loanId": "4",
      "amount": 5000000,
      "debt": 1000000,
      "tenure": 12,
      "status": {
        "statusId" : "4",
        "description" : "not yet paid off" || "belum lunas"
      },
      "createdAt": "2023-06-01T00:00:00Z",
```

	"updateAt": "2023-06-01T00:00:00Z"
	}
	]
	}
	}
	failed
	{
	"status": true,
	"code": "CU002",
	"message": null,
	"errorMessage": "Login failed",
	"errorData": {
	"errorTitle": "Login Failed",
	"errorDesc": "Please check your email/password"
	},
	"data": null
	}

Get List User Loans

Sequence Diagram



API Contract

Method	GET
Endpoint	{{host}}ms-loan /v1/list-loan? userId={userId}

Request	Header
	X-localization: {{LANGUAGE}}
	Authorization: Bearer jwt-token
	Content-Type:application/json
	x-api-Key: {{API_KEY}}
	channel: {{CHANNEL}}

	Body -

Response	**Success,Data Loan Found**
	{
	"status": true,
	"code": "LU001",
	"message": true,
	"errorMessage": null,
	"errorData": null,
	"data": {
	"loans": [
	{
	"loanId": "1",
	"amount": 5000000,
	"debt": 5000000,
	"tenure": 12,
	"status": {
	"statusId" : "1",
	"description" : "approved" || "diterima"
	},
	"createdAt": "2023-06-01T00:00:00Z",
	"updateAt": "2023-06-01T00:00:00Z"
	},
	{
	"loanId": "2",
	"amount": 15000000,
	"debt": 15000000,
	"tenure": 12,
	"status": {
	"statusId" : "2",
	"description" : "rejected" || "ditolak"
	},
	"createdAt": "2023-06-01T00:00:00Z",
	"updateAt": "2023-06-01T00:00:00Z"
	},

```
      {
        "loanId": "3",
        "amount": 5000000,
        "debt": 5000000,
        "tenure": 12,
        "status": {
          "statusId" : "3",
          "description" : "pending" || "menunggu"
        },
        "createdAt": "2023-06-01T00:00:00Z",
        "updateAt": "2023-06-01T00:00:00Z"
      },
      {
        "loanId": "4",
        "amount": 5000000,
        "debt": 0,
        "tenure": 12,
        "status": {
          "statusId" : "4",
          "description" : "paid off" || "lunas"
        },
        "createdAt": "2023-06-01T00:00:00Z",
        "updateAt": "2023-06-01T00:00:00Z"
      },
      {
        "loanId": "4",
        "amount": 5000000,
        "debt": 1000000,
        "tenure": 12,
        "status": {
          "statusId" : "4",
          "description" : "not yet paid off" || "belum lunas"
        },
        "createdAt": "2023-06-01T00:00:00Z",
        "updateAt": "2023-06-01T00:00:00Z"
      }
    ]
  }
}
```

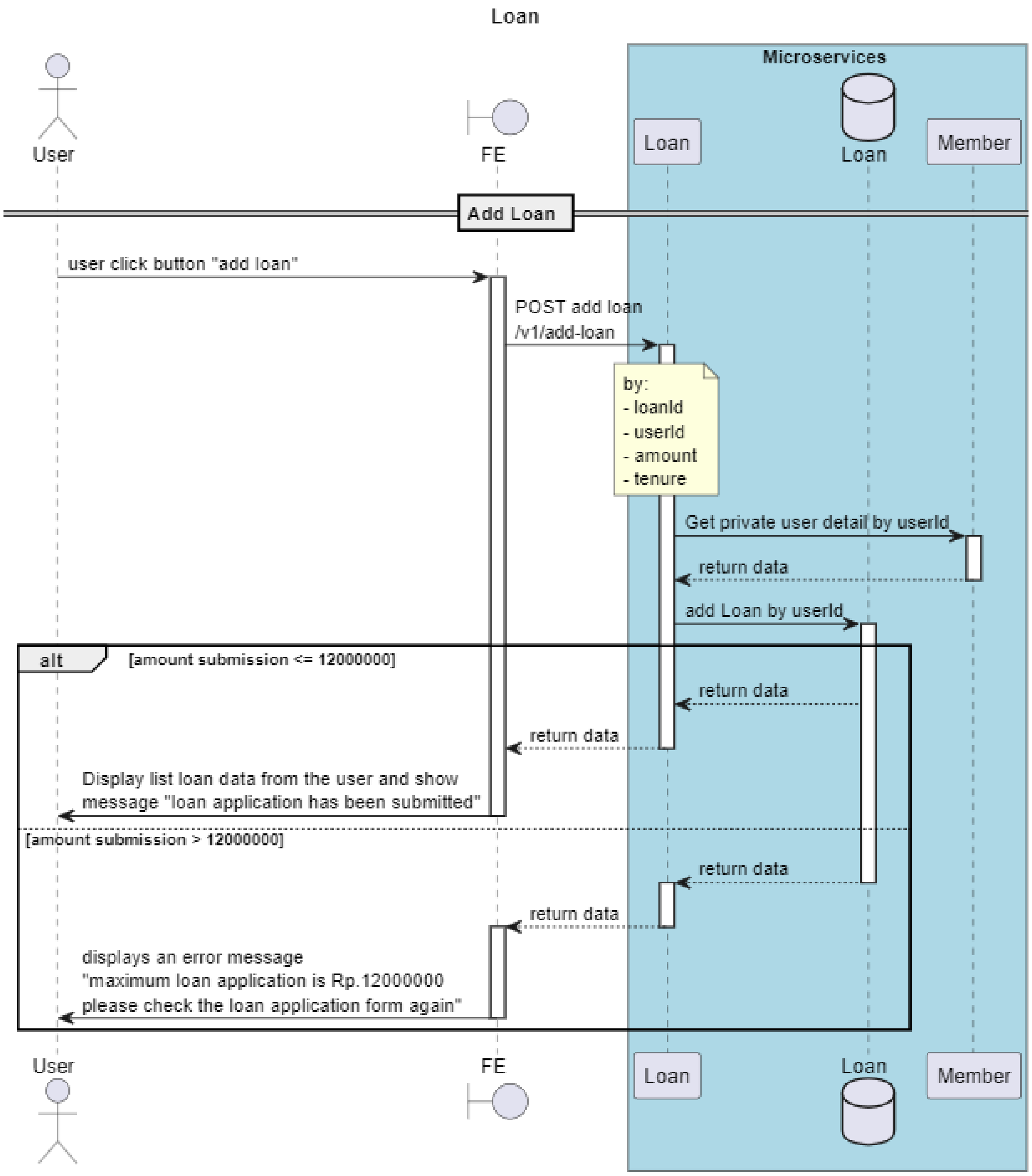
Success, no loan data

```
{
  "status": true,
  "code": "LU001",
  "message": true,
```

	"errorMessage": null,
	"errorData": null,
	"data": {
	"loans": []
	}
	}

Add Loan

Sequence Diagram



Loan
Microservices
User
FE
Loan
Loan
Member
Add Loan
user click button "add loan"
POST add loan
/v1/add-loan
by:
- loanId
- userId
- amount
- tenure
Get private user detail by userId
return data
add Loan by userId
alt
[amount submission <= 12000000]
return data
return data
Display list loan data from the user and show
message "loan application has been submitted"
[amount submission > 12000000]
return data
return data
displays an error message
"maximum loan application is Rp.12000000
please check the loan application form again"
User
FE
Loan
Loan
Member

API Contract

Method	POST
Endpoint	{{host}}ms-loan/v1/add-loan

Request	Header X-localization: {{LANGUAGE}} Authorization: Bearer jwt-token Content-Type:application/json x-api-Key: {{API_KEY}} channel: {{CHANNEL}}h

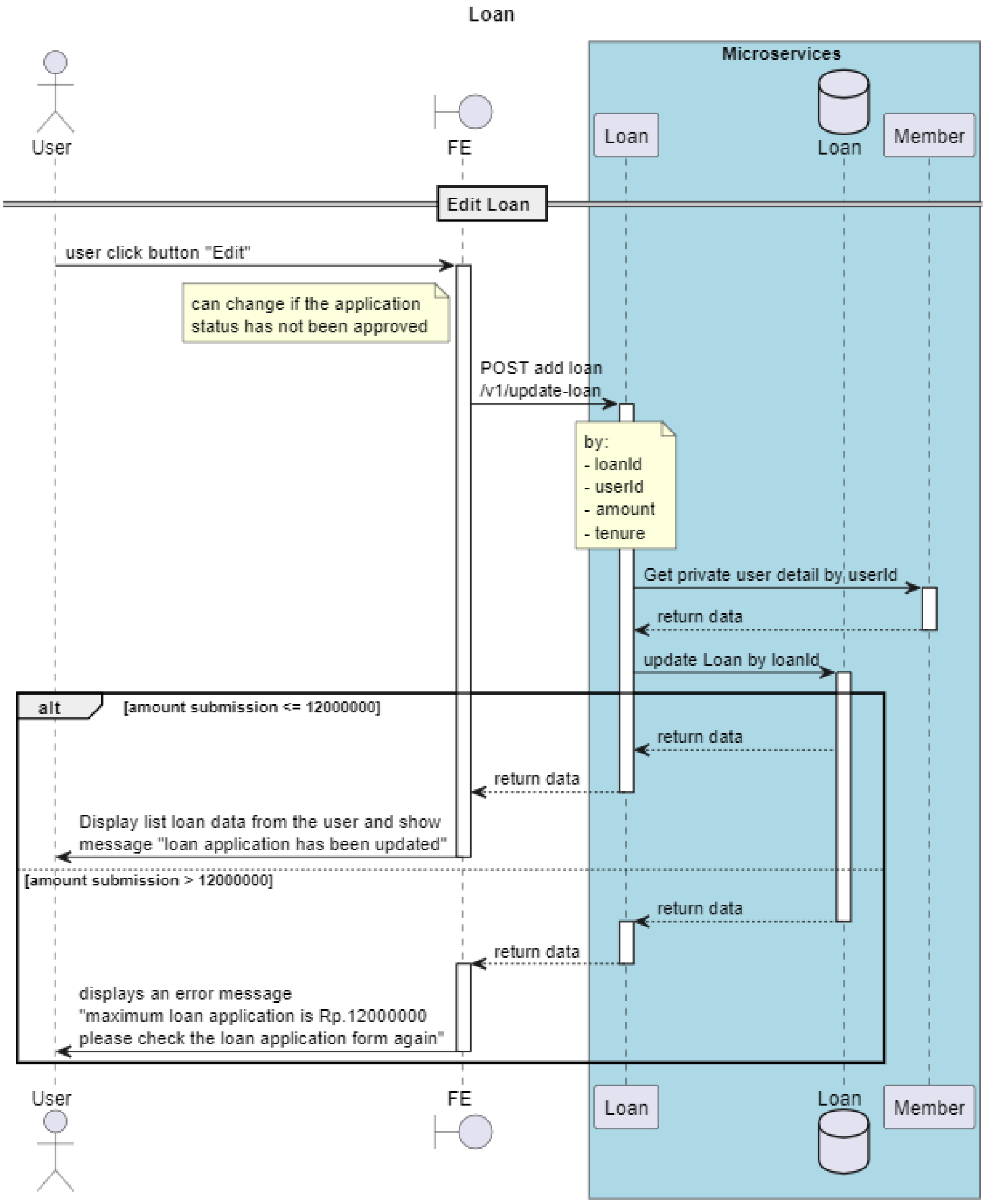
	Body
	{
	"member":{
	"userId":1,
	"nama": "sukijan",
	"email": "user@example.com",
	"phoneNumber": "08123456789",
	"password": "password123",
	"photo": "base64-encoded-image",
	"ktp": "base64-encoded-image"
	},
	"loan" : {
	"loanId": 1,
	"amount": 10000000,
	"tenure": 12,
	},
	}

Response	**success**
	{
	"status": true,
	"code": "AL001",
	"message": true,
	"errorMessage": null,
	"errorData": null,
	"data": {
	"message": "Loan application submitted",
	"loan" : {
	"loanId": 1,
	}
	}
	}
	failed
	{
	"status": true,
	"code": "LU002",
	"message": null,
	"errorMessage": "Loan application not submitted",
	"errorData": {

	"errorTitle": "Loan application not submitted", "errorDesc": "Please check your loan application form" }, "data": null }

Update Loan

Sequence Diagram


Loan
Microservices
User
FE
Loan
Loan
Member
Edit Loan
user click button "Edit"
can change if the application status has not been approved
POST add loan /v1/update-loan
by:
- loanId
- userId
- amount
- tenure
Get private user detail by userId
return data
update Loan by loanId
alt
[amount submission <= 12000000]
return data
return data
Display list loan data from the user and show message "loan application has been updated"
[amount submission > 12000000]
return data
return data
displays an error message "maximum loan application is Rp.12000000 please check the loan application form again"
User
FE
Loan
Loan
Member

API Contract

Method	POST
Endpoint	{{host}}ms-loan/v1/update-loan

Request	Header
	X-localization: {{LANGUAGE}}
	Authorization: Bearer jwt-token
	Content-Type:application/json
	x-api-Key: {{API_KEY}}
	channel: {{CHANNEL}}

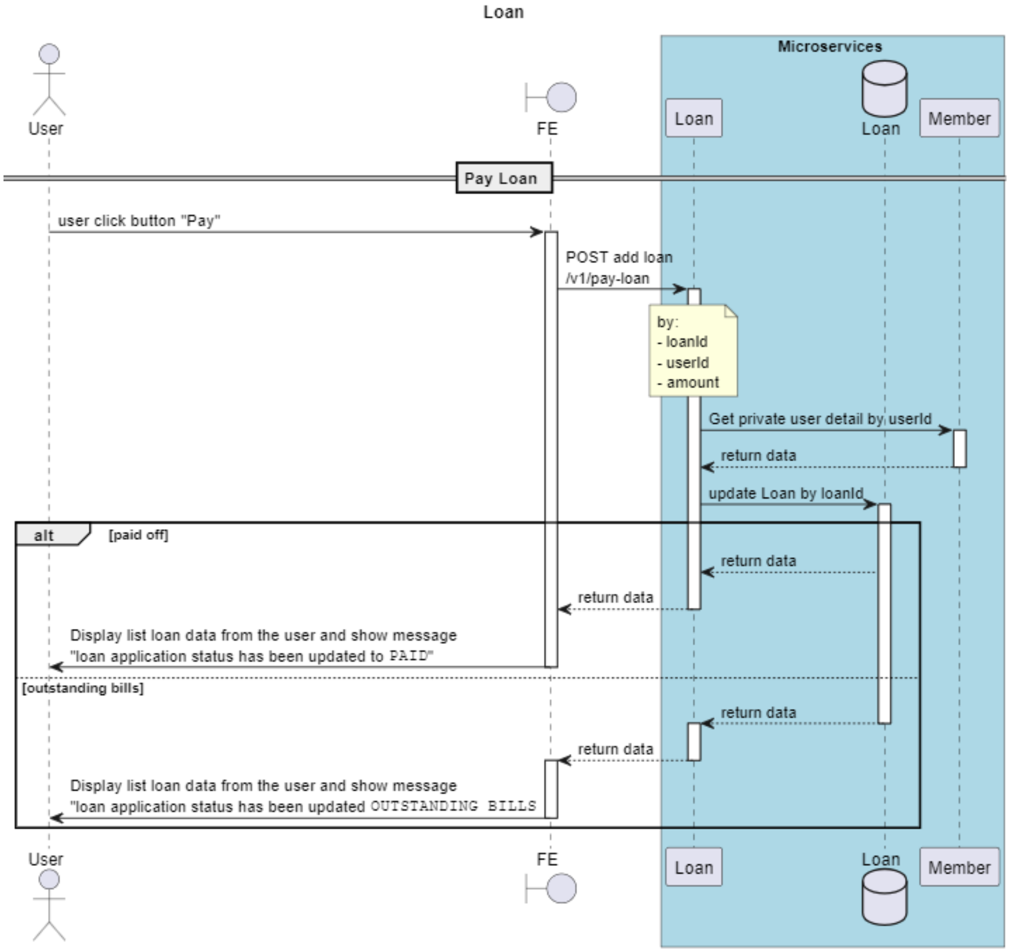
	Body
	{
	"member":{
	"userId":1,
	"nama": "sukijan",
	"email": "user@example.com",
	"phoneNumber": "08123456789",
	"password": "password123",
	"photo": "base64-encoded-image",
	"ktp": "base64-encoded-image"
	},
	"loan" : {
	"loanId": 1,
	"amount": 10000000,
	"tenure": 12,
	}
	}

Response	**success**
	{
	"status": true,
	"code": "AL001",
	"message": true,
	"errorMessage": null,
	"errorData": null,
	"data": {
	"message": "Loan application updated",
	"loan" : {
	"loanId": 1,
	"amount": 10000000,
	"tenure": 12,
	}
	}
	}
	failed
	{
	"status": true,
	"code": "LU002",
	"message": null,

	"errorMessage": "Loan application not updated",
	"errorData": {
	"errorTitle": "Loan application not submitted",
	"errorDesc": "Please check your loan application"
	},
	"data": null
	}

Pay Loan

Sequence Diagram


Loan
Microservices
User
FE
Loan
Loan
Member
Pay Loan
user click button "Pay"
POST add loan
/v1/pay-loan
by:
- loanId
- userId
- amount
Get private user detail by userId
return data
update Loan by loanId
alt
[paid off]
return data
return data
Display list loan data from the user and show message
"loan application status has been updated to PAID"
[outstanding bills]
return data
return data
Display list loan data from the user and show message
"loan application status has been updated OUTSTANDING BILLS
User
FE
Loan
Loan
Member

API Contract

Method	POST
Endpoint	{{host}}ms-loan/v1/pay-loan

Request	Header
	X-localization: {{LANGUAGE}}
	Authorization: Bearer jwt-token
	Content-Type:application/json
	x-api-Key: {{API_KEY}}
	channel: {{CHANNEL}}

	Body
	{
	"member":{
	"userId":1,
	"nama": "sukijan",
	"email": "user@example.com",
	"phoneNumber": "08123456789",
	"password": "password123",
	"photo": "base64-encoded-image",
	"ktp": "base64-encoded-image"
	},
	"loanId": 1,
	"amount": 10000000,
	"tenure": 12
	}

Response	**success**
	{
	"status": true,
	"code": "PL001",
	"message": true,
	"errorMessage": null,
	"errorData": null,
	"data": {
	"message": "Loan application paid, debt loan amount 1000000 with tenure 10 months",
	"loanId": "2"
	}
	}
	failed
	{
	"status": true,
	"code": "PU002",
	"message": null,
	"errorMessage": "Loan application not paid",
	"errorData": {
	"errorTitle": "Loan application not paid",
	"errorDesc": "Please check your loan application"
	},
	"data": null
	}

4. Buatlah detail design untuk screen behavior dari mobile apps berdasarkan screen flow diatas.

Detail design

No.	Name Screen	Behavior
1.	Welcome Screen	• Display buttons for "Register" and "Login". • Navigate to Registration Screen or Login Screen based on user selection.
2.	Registration Screen	• Collect user data (email, phone number, password, photo, KTP). • Validate inputs. • On submission, call User Registration API. • Display success or error message based on API response.
3.	Login Screen	• Allow login via password or biometric. • On submission, call User Login API. • Store JWT token on successful login. • Navigate to Dashboard.
4.	Dashboard	• Fetch user loans using Get User Loans API. • Display outstanding loans and monthly bills. • Provide option to apply for a new loan. • Navigate to Loan Application Screen on selection.
5.	Loan Application Screen	• Collect loan amount and tenure from the user. • Validate inputs (e.g., maximum amount, maximum tenure). • On submission, call Apply for Loan API. • Display success or error message based on API response.
6.	Loan Status Screen	• Display loan application status (approved, rejected, pending). • Fetch loan status using Get Loan Status API.
7.	Notifications	• Display notifications for loan approval/rejection. • Fetch notifications from the server and display them to the user.