AMIGO
Developer Documentation

Authors
Florian Hintermeier
Gusztáv Ottrubay
Anamaria Skočić
Matthias Bernad

Intro

In concern of global OpenSource community, this doc is kept in English, even though German would be sufficient.

Developer's Setup

amigo-server

See https://gitlab.com/ossi-austria/amigo-server/-/blob/develop/README.md

Requirements

- An IDE, IntellijIDEA by Jetbrains can be recommended
- installed docker tools, or docker-desktop for windows

Environment Variables

JITSI_JWT_APP_SECRET=***;
JITSI_JWT_APP_ID=amigo-platform-dev;
JITSI_ROOT_URL=https://***/

Setup

1. Clone the project and checkout *develop* branch

```
git clone git@gitlab.com:ossi-austria/amigo-server.git
cd amigo-server
git checkout develop
```

2. Open the project in favorite IDE (we use IntelliJ IDEA)

3. Start services and database needed for development

```
docker-compose up
```

Attention:
As the development Environment will only rely on POSTGRES **within** docker, you need to add the variables defined in system.env to the environment which runs the spring boot server. Do that a) as ENV VARs or better b) in IDE running config (all marked Env Vars above have to be provided

4. Start the "RestApplication" via Intellij with "**dev**" spring env profile active

5. Add firebase-config

Firebase is needed for FCM to support push notifications. Additional configs can be provided, in a $ROOT/configs directory which is ignored by git and NOT compiled into the app, but merely used as runtime config.

- Ask someone for more information
- Add **firebase-service-account.json** into that directory
- do not check-in server credentials in your (public) repository

amigo-box

See https://gitlab.com/ossi-austria/amigo-box/-/blob/develop/README.md

As amigo-box is a "typical" Android App, nothing special is needed for setup.

- Have JDK 11 ready (bundled with current Android Studio)
- Use a Tablet, a (Tablet) Emulator, or a larger Phone in landscape mode for testing
- **Only important setup step:** add Amigo-Server to your local.properties (should be gitinored). See https://gitlab.com/ossi-austria/amigo-box/-/blob/develop/local.properties.example
- Run gradlew assembleDebug
- Start App and login with *your* registered test user on *your* server

amigo-app

AmigoApp is created in Flutter, and focussed for Android. For Development, just prepare the development environment:

https://docs.flutter.dev/get-started/install

REST-Api documentation

You can either create the REST-Api on your own with gradle, or use our offical REST-Api documentation:

- https://ossi-austria.gitlab.io/amigo-server/develop/rest-api.html
- http://amigo-dev.ossi-austria.org:8080/swagger-ui.html
- http://amigo-dev.ossi-austria.org:8080/v2/api-docs

With Swagger-UI you can view and test: http://amigo-dev.ossi-austria.org:8080/swagger-ui.html

- login with your Test user, via auth-api login (click on "Try out") { "email": "string", "password": "string" }
- remember your data: **personId**, **accessToken**, eventually groupId
- On every Request, provide **personId** as **Amigo-Person-Id** and **accessToken** as **Authorization**

Create OpenApi client from openApi

Note: Works with openApi v2 currently.

docker run --rm -v "${PWD}:/local" openapitools/openapi-generator-cli generate -i http://192.168.0.20:8080/v2/api-docs -g dart -o /local/out/dart

https://springframework.guru/spring-boot-restful-api-documentation-with-swagger-2/

“Production” environment

As amigo is still in development, we advise against using it for “production-ready” environments, because it just isn’t. For demo and development environments, we created a docker-compose stack, which can be started easily.

Similar to “amigo-server setup”, you need docker and docker-compose installed on your Linux system.

1. Install docker and docker-compose

https://docs.docker.com/compose/install

2. Setup the proper environment for docker

Copy "system.env.default" to **system.env** and **change SECRETs and DB passwords** before use in a production environment!

Set **at least** the following variables, JITSI_JWT_APP_SECRET should have 32 chars!

```
AMIGO_ACCESS_TOKEN_SECRET=547fewtabd4w68b4w6
AMIGO_REFRESH_TOKEN_SECRET=547fewtabd4w68b4w7

JITSI_JWT_APP_ID=jitsi
JITSI_JWT_APP_SECRET=bbbbvvxxywqqqdddccsssxgasdfzaaaa
JITSI_ROOT_URL=https://amigo-dev.ossi-austria.org/
```

3. Initialise docker container

This will start the amigo-platform and postgres services with the applied system.env

```
docker-compose -f docker-production.yml up -d
```

After updating the server, via git pull, you might restart everything:

```
#!/bin/bash
cd amigo-platform
docker-compose pull
docker-compose up -d
docker-compose ps
docker-compose logs
```

Setting up Jitsi with JWT auth for amigo-server

See: https://github.com/jitsi/docker-jitsi-meet

We use a plain demo jitsi docker server for testing reasons. The only important thing is the configuration of JWT:

- do complete guided steps from docker jitsi setup first!
- in copied ".env" file, make sure you have the following:

```
# Enable authentication
ENABLE_AUTH=1
ENABLE_GUESTS=0
AUTH_TYPE=jwt
# JWT authentication
JWT_APP_ID=jitsi
JWT_APP_SECRET=*secret
JWT_TOKEN_AUTH_MODULE=token_verification
# (Optional) Set asap_accepted_issuers as a comma separated list
JWT_ACCEPTED_ISSUERS=jitsi,amigo-platform-dev,AMIGO-PLATFORM
# (Optional) Set asap_accepted_audiences as a comma separated list
JWT_ACCEPTED_AUDIENCES=jitsi,amigo-platform-dev,AMIGO-PLATFORM
```