

---

# Seven23 Documentation

*Release 1.2.0*

**Sébastien BARBIER**

**Sep 07, 2023**



---

## Contents

---

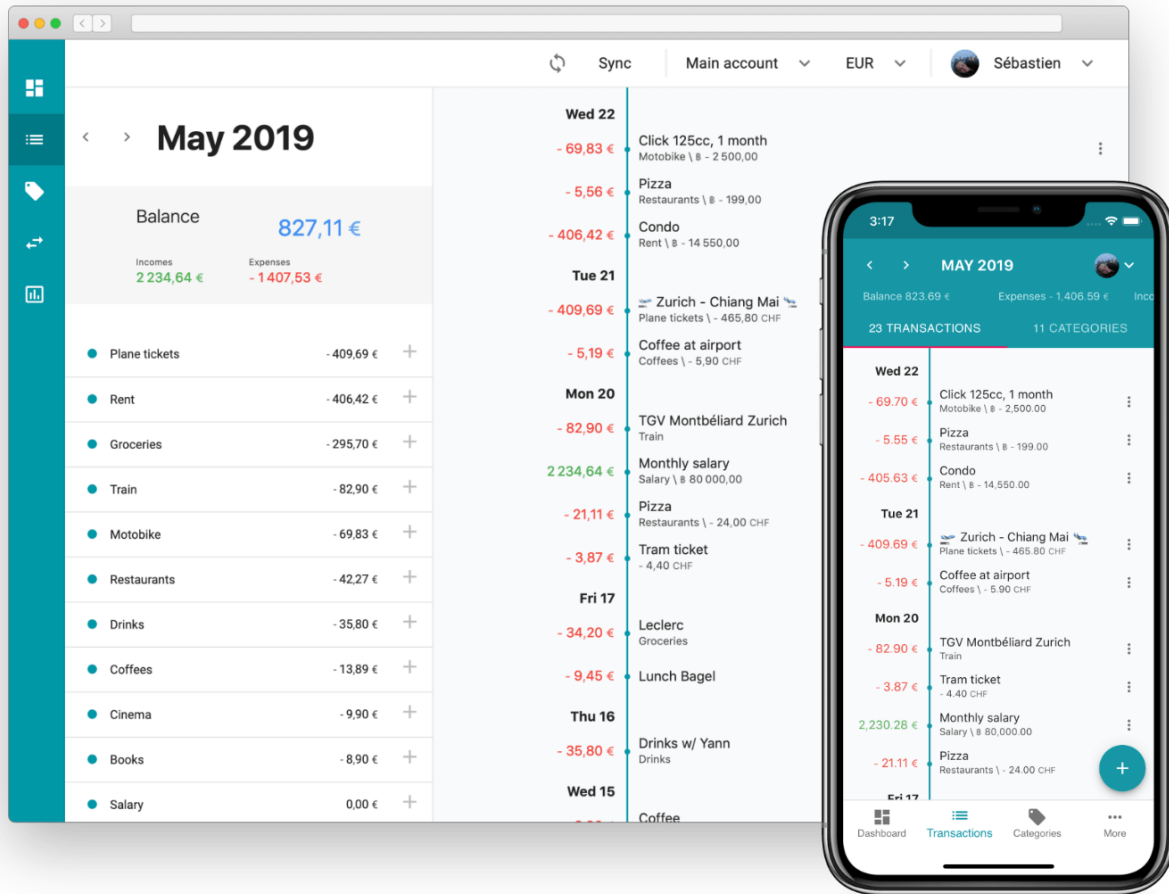
<b>1</b>	<b>Summary</b>
----------	----------------

<b>3</b>
----------



Fully manual budget app to track your expenses. Completely opensource, with privacy by design.

- [Seven23 official website](#)
- [Web application](#)
- [Online documentation](#)
- [Issue tracker](#)
- [Code repository](#)



**Note:** App can run on device, **no account needed**, but need a server instance to sync and backup your data. Server code is also open source and available as a separate repository: [seven23\\_server](#).



# CHAPTER 1

---

## Summary

---

## 1.1 Installation

```
git clone git@github.com:sebastienbarbier/seven23.git
```

### 1.1.1 npm install

```
npm install
```

### 1.1.2 Run devserver

```
npm start
```

## 1.2 Exchange rate

Seven23 webapp allows to switch currency at any point, and manage to estimate the price for each transaction. Sound great, but this is implemented in a dummy way and **do not guaranty exactitude** on it.

### 1.2.1 Rates

Exchange rate is based on the `change` model. Each change has a date, two currency and two amount involved in the transaction. For each change, we can calculate the exact exchange rate.

One easy way to do is to generate what we called a **changeChain** which is technically an array of change order by date. We loop on the array and for each change, save the current rate for each currency so this way for any date, we only need to access the latest change item from the chain based on its date to have all current exchange rate.

To calculate a transaction price, we look at its date, and get the earliest past change event for .

### 1.2.2 Second degree rates

One use-case we wanted to handle is the *second degree rates*. If you have euros, exchange them with swiss franc, then exchange swiss francs with thai baths, we wanted to be able to estimate an exchange rate between euros and thai baths.

To do so, we loop on the *changeChain* and for each one, we calculate what would be a two jump with the exchange we are browsing on. **This might give very different result based on how we browse the chain and is a known issue, but is better than nothing.**

---

**Note:** In app, we display second degree calculation with an average sign in front of each prive.

---

### 1.2.3 First rates

A last situation is about defining an exchange rate before any change event occurred. To solve this, we store in a *firstRated* object each rate to use as default. This one is generated **first time a rate or a secondDegree rate is changed** and used as a default value if needed.

## 1.3 End-to-end encryption

Seven23 webapp implement **end-to-end encryption** to protect user's privacy. Server only store encrypted blob and cannot access any details about encrypted components.

Encryption is performed using the *js-jose* library in **JSON Web Encryption (JWE)** format with A128CBC-HS256 algorithm and A256KW encryption key.

Encryption key currently use a md5 value of your password, so losing it means you cannot access your data anymore.

**Warning:** Data are encrypted/decrypted on every server request, but are stored unencrypted on your browser meaning anyone with a physical access to your machine can read those if you are still logged in.

## 1.4 Build and deployment

### 1.4.1 Build static files from source

This process bundles js, css, and html as static stand-alone files.

```
npm run build
```

Those can be served by any static file server without any dependencies.



## 1.4.2 Deploy using Docker

The project includes Docker and docker-compose files for deployment.

Build and run it directly with docker-compose:

```
docker-compose up
```

Or build the container...

```
docker build -t seven23 .
```

...and run it:

```
docker run -p 80:80 -ti seven23
```

## 1.5 Documentation

This official documentation is maintained in GitHub. The docs folder contains the documentation sources in reStructuredText format. And you can generate the docs locally with:

```
make docs
```

Output is written at docs/\_build/html/index.html.

## 1.6 Constribution

### 1.6.1 How to contribute

Thanks for your interest in contributing to Seven23 webapp!

---

**Note:** We love community feedback and are glad to review contributions of any size - from typos in the documentation to critical bug fixes - so don't be shy!

---

### 1.6.2 Report bugs

Report bugs at <https://github.com/sebastienbarbier/seven23/issues/new>

If you are reporting a bug, please include:

- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

### 1.6.3 Fix bugs

Check out the [open bugs](#) - anything tagged with the **easy-pick label** could be a good choice for newcomers.

### 1.6.4 Implement features

Look through the GitHub issues for features. Anything tagged with **enhancement** is open to whoever wants to implement it.

### 1.6.5 Write documentation

Seven23 could always use more documentation, whether as part of the official docs, in docstrings, or even on the Web in blog posts, articles, and such.

This official documentation is maintained in GitHub. The docs folder contains the documentation sources in reStructuredText format. And you can generate the docs locally with:

```
make docs
```

Output is written at docs/\_build/html/index.html.

We obviously accept pull requests for this documentation, just as we accept them for bug fixes and features!