ERP & CRM

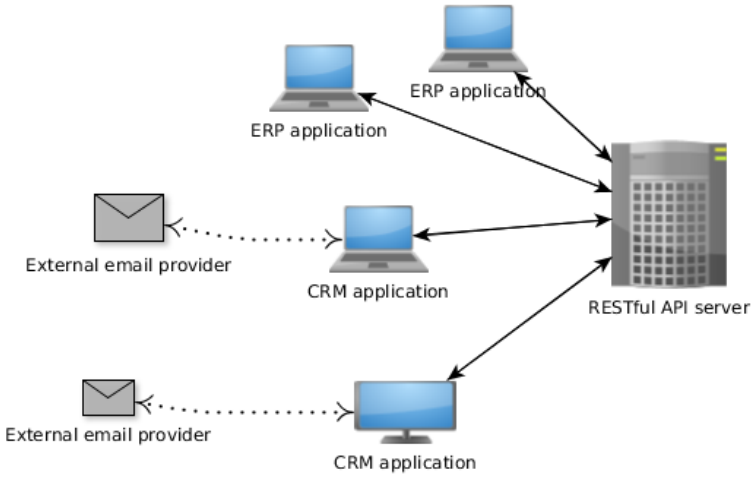
Paweł Płatek, Marcin Blacharski, Grzegorz Puczkowski, Agnieszka Barszcz

January 24, 2018

1 Introduction

1.1 General description

Enterprise resource planning and Customer Relationship Management systems. System is composed of two frontend desktop applications (for crm and erp) and backend server with RESTful API web service (common for both apps). Desktop applications use external email providers to send, receive and store emails.



1.2 Functionalities

- General
  - crm, erp and admin employees
- ERP
  - Companies / customers management
  - Orders management
  - Proformas generation
  - Warehouse
  - Delivery costs
  - Reports (sum of orders for each employee in time period, articles sold)
- CRM
  - Companys / customers / contacts management

- Tasks
- Meetings
- Email communication

1.3 Technology and licences

Backend

- Spring Boot (Apache License 2.0)
- Sprint Data REST (Apache License 2.0)
- Spring Data JPA (Apache License 2.0)
- Spring REST Assured (Apache License 2.0)
- Maven (Apache License 2.0)
- Swagger / Springfox (Apache License 2.0)
- Postgresql (PostgreSQL License)

REST API was created in swagger (openAPI) format. Swagger made it possible to partially define the system without use of specific technology. Moreover, defined api at the beginning allowed developers to work on client applications and server parallely. Springfox (part of swagger project) is plugin for spring to document api inside source code.

Server is a Spring Boot application. Main adventage is that it's self-contained (contains embedded tomcat servlet, runs from single jar file) which make it easy to ship, deploy and possibly containerize. Another adventage is consistency with frontend applications (also made in java). Rest of the features (Data JPA / ORM, auto CRUD, unit tests framework) can be found in other technologies/frameworks.

PostgreSQL was chosen mainly because it's open source and efficient.

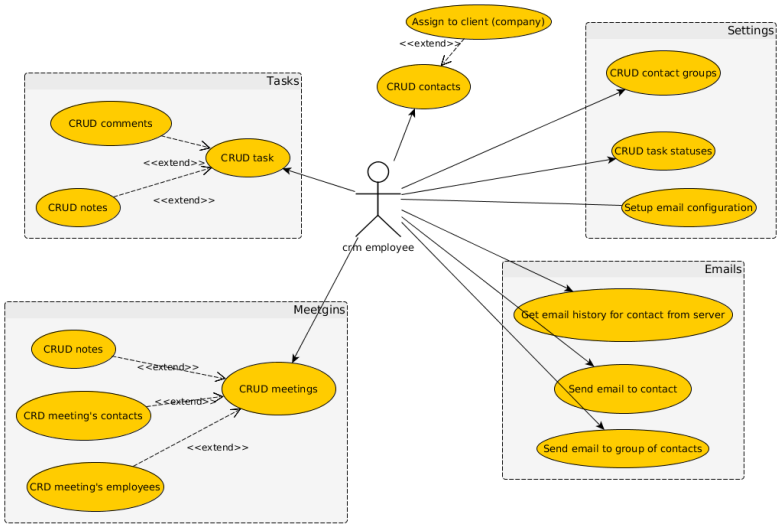
Frontend

- JavaFX
- CSS
- javamail - mail client (CDDL and GPLv2+CE)
- HTML
- iText - pdf generating (MPL/GPL license)

1.4 Use cases and Entity Relationship Diagram

Employees management

ERP



CRM

2 Backend

Swagger generated, REST api

2.1 Overview

This server was generated by the swagger-codegen project.

The underlying library integrating swagger to SpringBoot is springfox

Default database is postgres.

2.2 Setup

First, check configuration in src/main/resources/application.properties and application-test.properties:

- server.port - default 8080
- spring.datasource.url - default localhost:5432/io
- spring.datasource.username / password - for database
- security.oauth2.client.id / client-secret - for basic auth in Oauth (these are hardcoded into clinet applications, do not change)
- security.default.admin.user / mail / password - for auth

Second, create postgres databases (io and test_io by default). Installation guides here.

2.3 Release

```
java -jar release/io-api-server-1.0.0.jar
```

Probably will not work, beacause of database settings. Please build as described below.

2.4 Build

```
mvn package
```

```
# or without tests
mvn package -Dmaven.test.skip=true
```

2.5 Run

```
java -jar ./target/io-api-server-1.0.0.jar
```

2.6 Tests

To run integration tests:

```
mvn integration-test -P integration
```

To run unit tests:

```
mvn test
```

2.7 Documentation

API is documented with swagger. After starting server you can find documentation at http://ip:port/api/swagger-ui.html.

2.8 Authentication

Access to API is restricted to users authenticated with token. To generate token send post request to http://ip:port/api/oauth/token with basic auth configured in resources/application.properties in security.oauth2 section and with following parameters: grant_type=password, username, password. For example:

```
curl -X POST "http://client_id:client_secret@localhost:8080/api/oauth/token"
-H "accept: application/json" -d grant_type=password -d username="" -d password=""
```

To use the token, send requests with "Authorization: Bearer token" header.

There are three roles for employees: admin, erp and crm. All of them have access to /api endpoint. Admin have access to all api endpoints and can change every employee's data. Employees with roles erp or crm are restricted to /api/erp/ or /api/crm/ endpoints and can change only own data.

Default users credentials can be found in application.properties.

Remember to change them and default basic auth credentials.

3 Frontend

3.1 Overview

The frontend part communicates with backend using REST http connection.

It is created in JavaFX with use of Cascade stylesheets for JavaFX. GUI allows for basic CRUD operations on objects, with possibility to export Orders and Proformas to PDF and send them via mail. The mail client can send messages to many recipients (separated by , or ;), both: visible and CC, with pre-defined attachment like orders or proformas and with any files chosen from file system.

The tasks and notes (in Tasks and Meetings view) can be displayed in custom colors.

User can invite others to tasks.

3.2 Setup

First set in config.json server address and port number following a pattern:

```
{
  "address" : "localhost",
  "port" : "8080"
}
```

After starting application in Settings/Libraries add all dictionary values.

To use e-mail complete the settings in tab "mail" and click "save". Fill profile data. (Optional) In "default values" set default paths, company logo etc.

3.3 Release

```
java -jar release/erp-1.0.0.jar
```

3.4 Build

```
mvn install
```

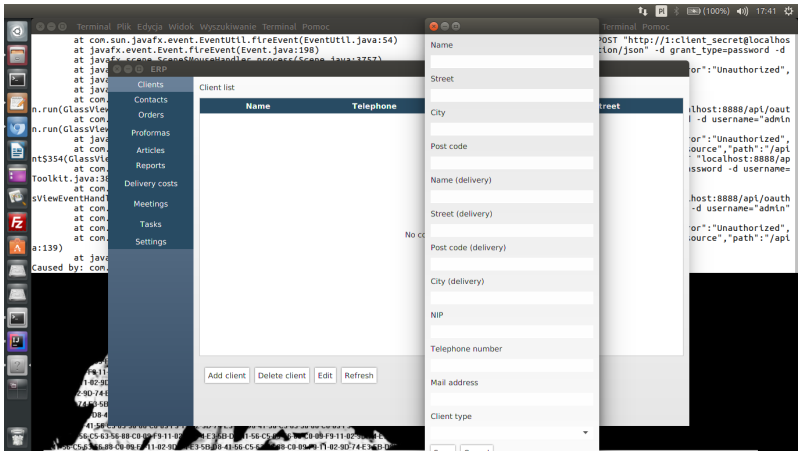
3.5 Run

```
java -jar ./target/erp-1.0.0.jar
```

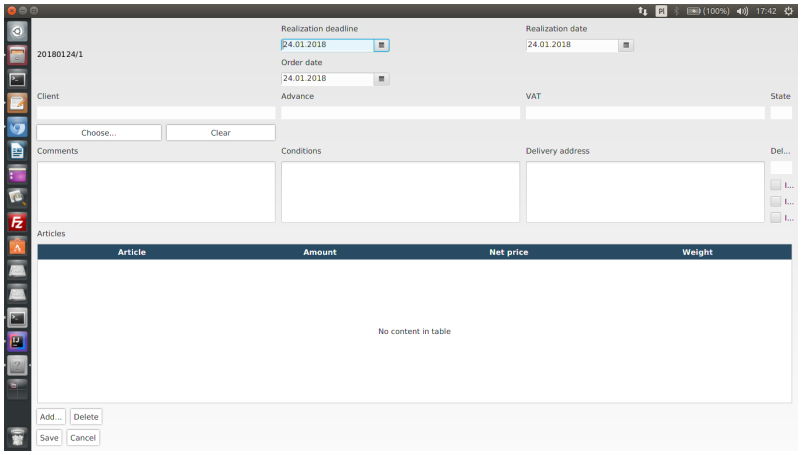
3.6 Authentication

Login and password from database.

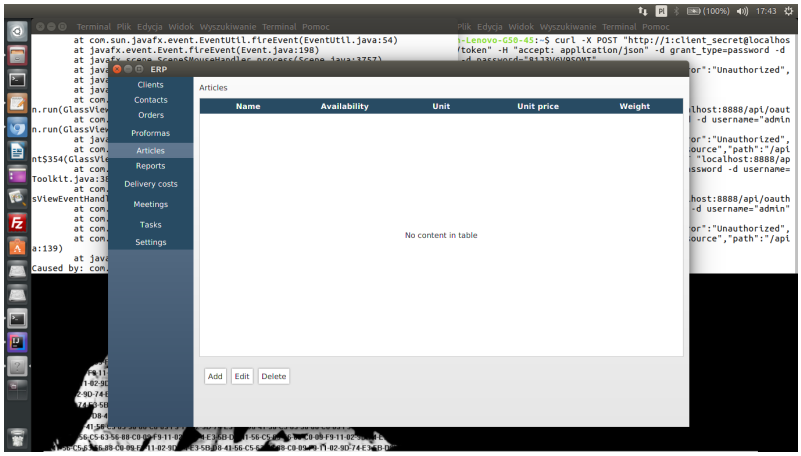
3.7 Documentation



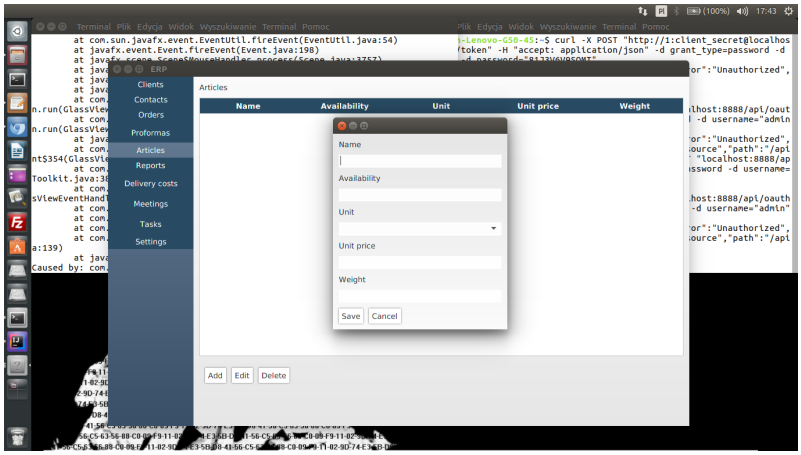
adding client



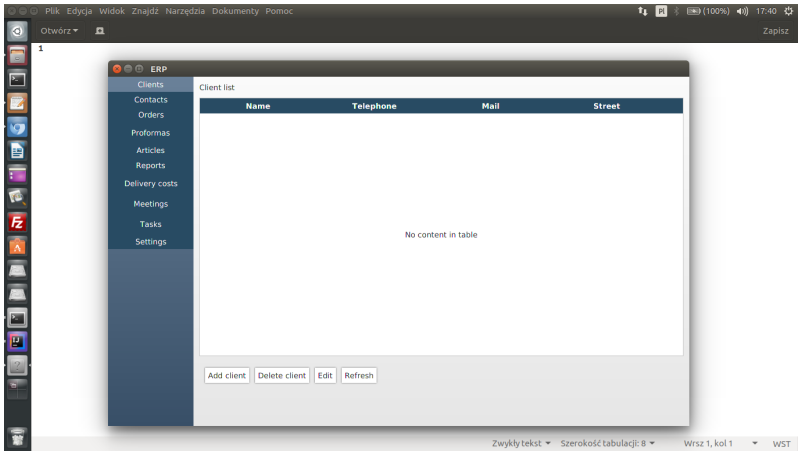
adding order



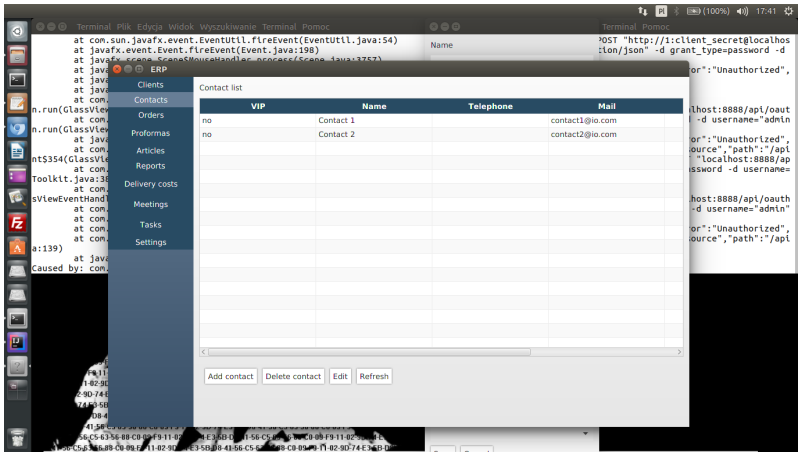
articles



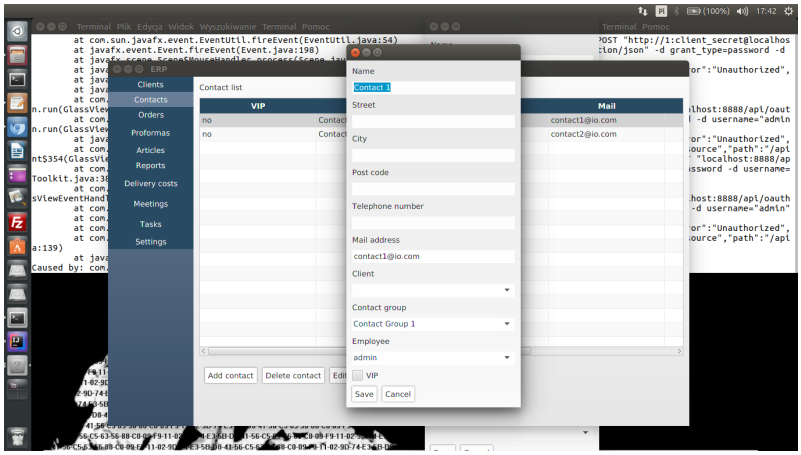
adding articles



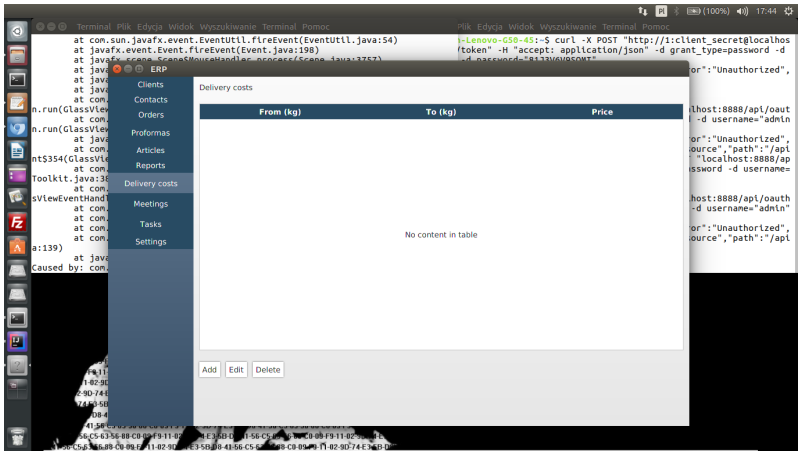
clients



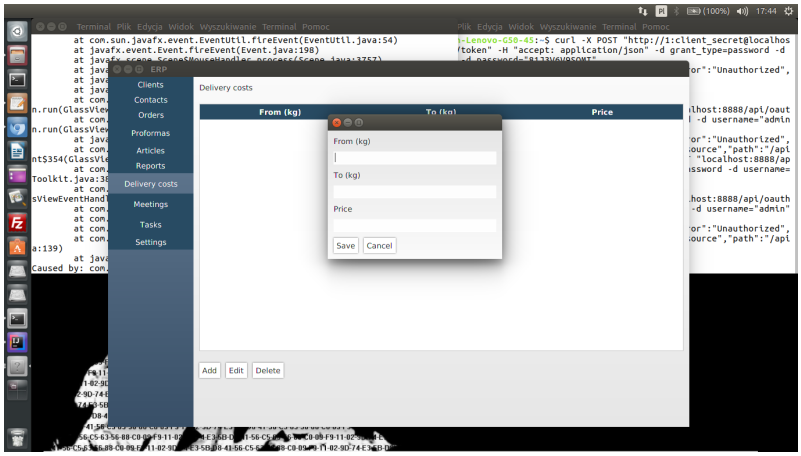
contacts



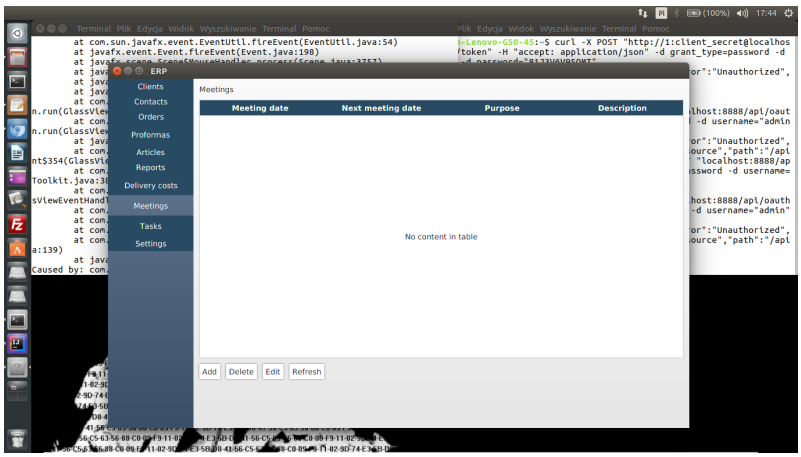
editing contact



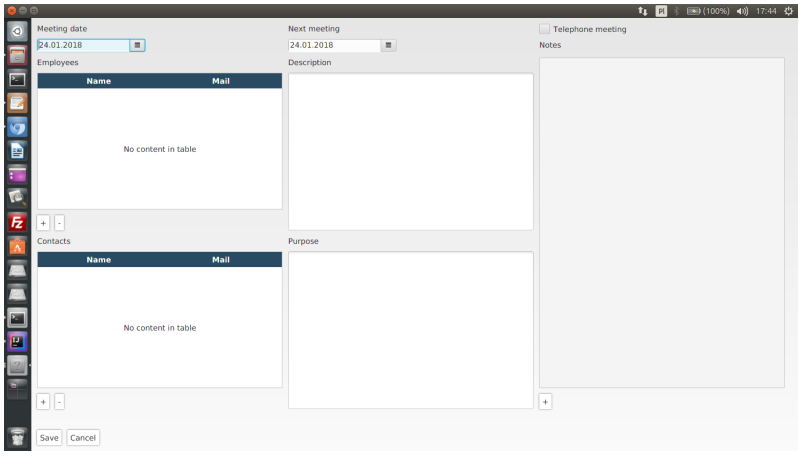
delivery costs



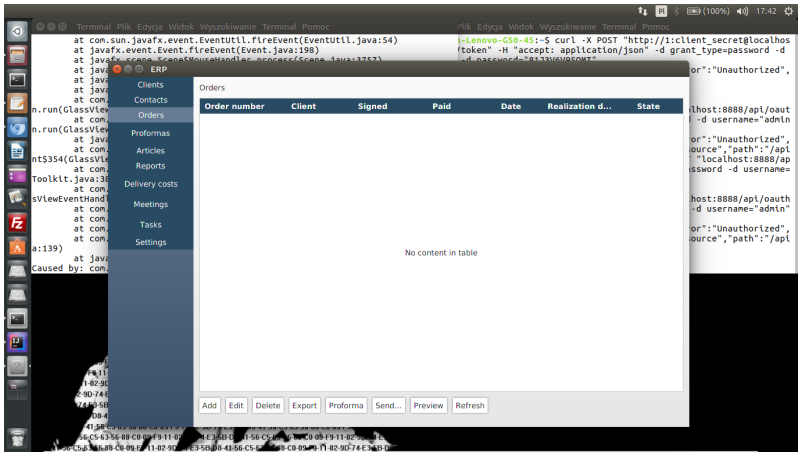
adding delivery costs



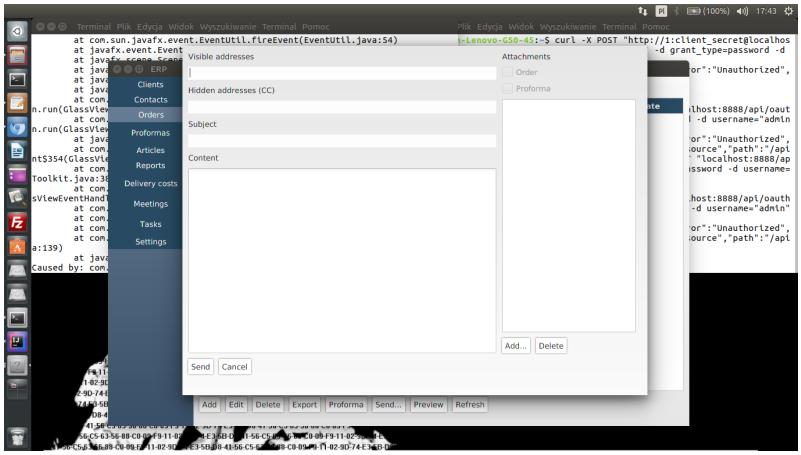
meetings



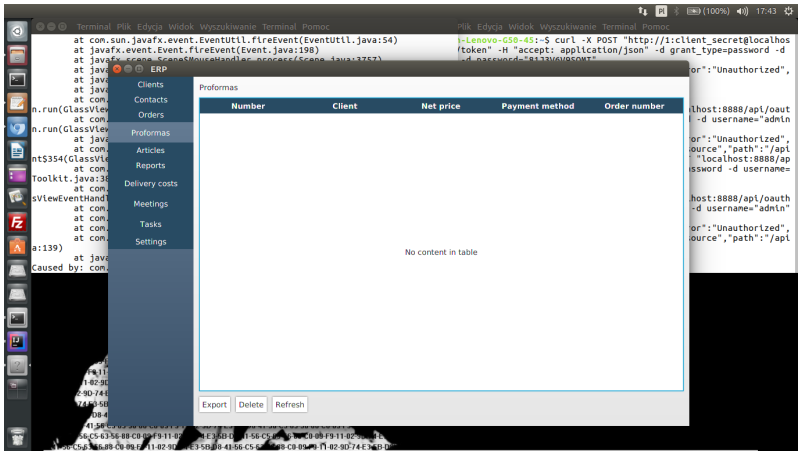
adding meetings



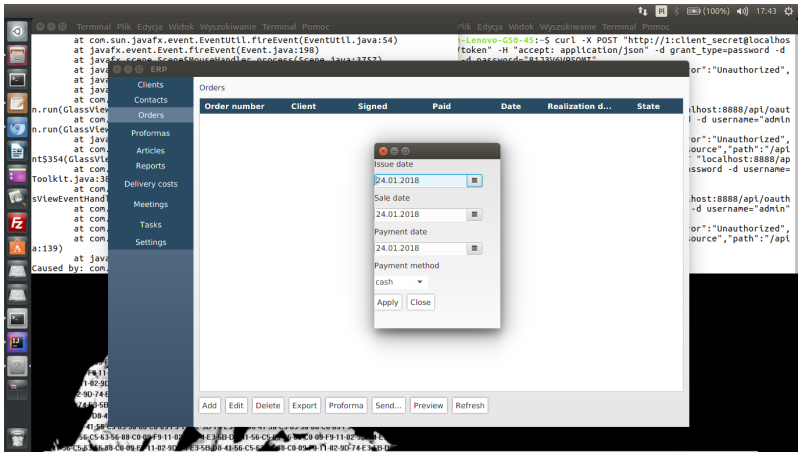
orders



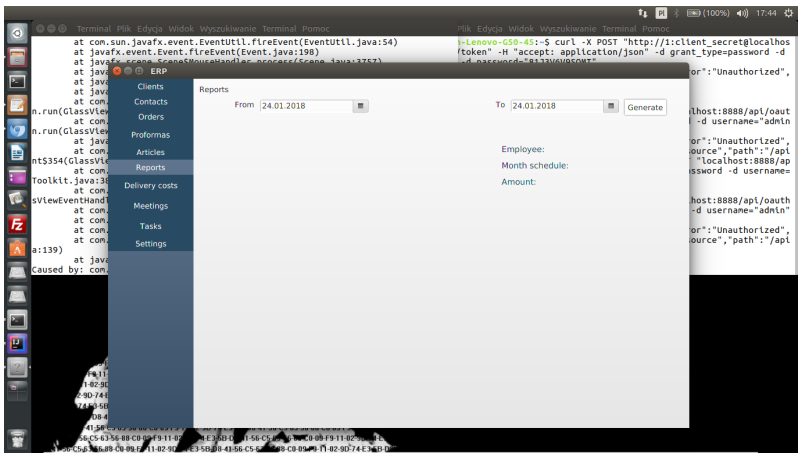
sending orders



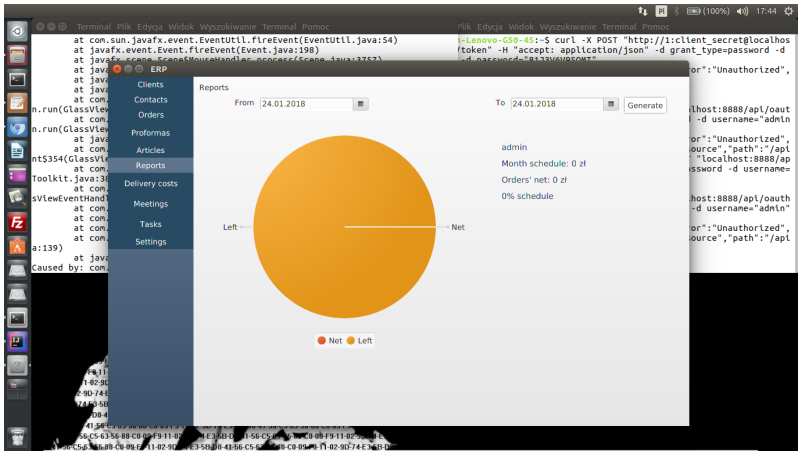
proformas



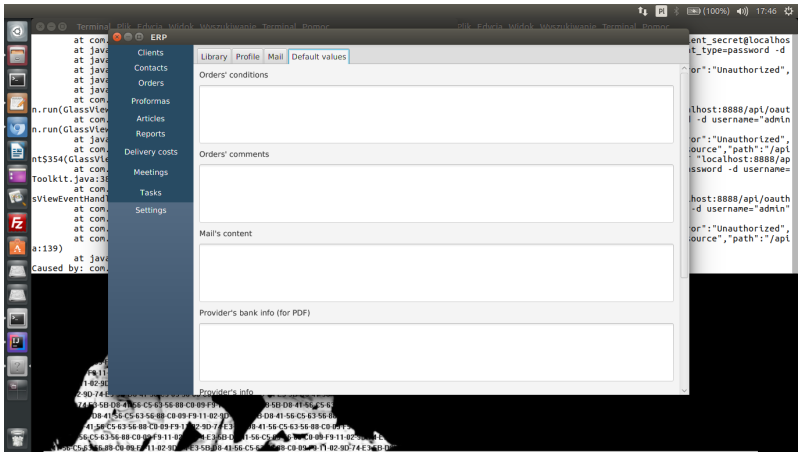
generating proforma



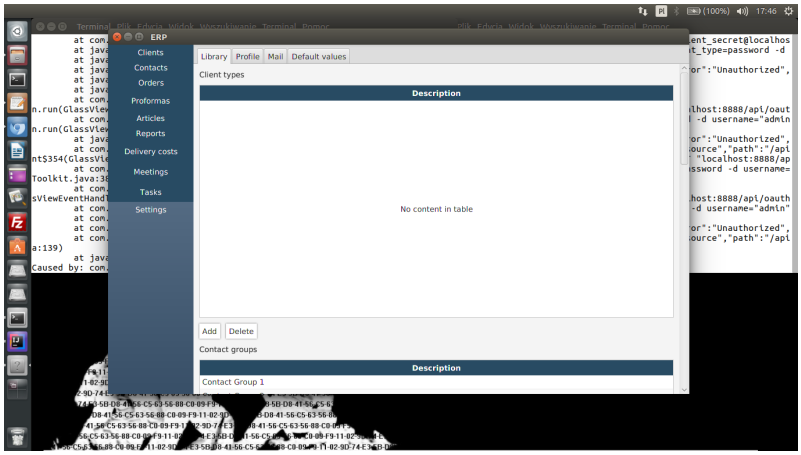
reports



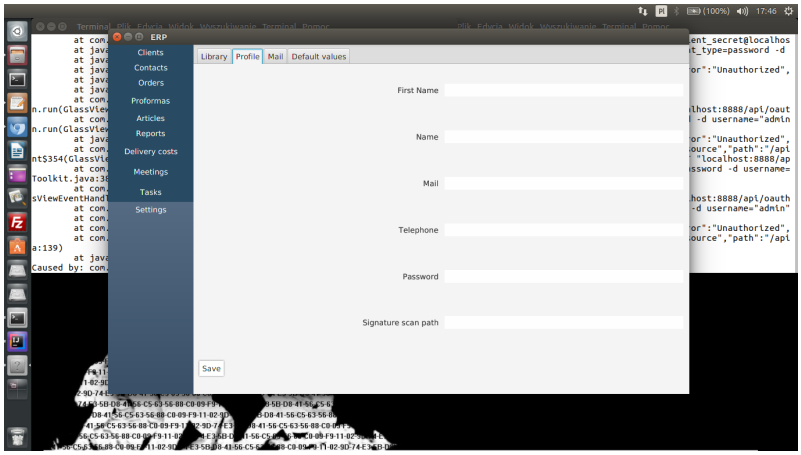
generating reports



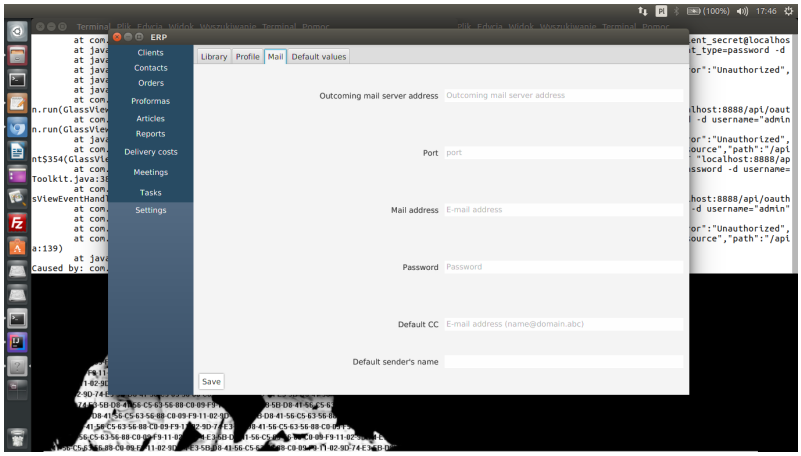
setting default values



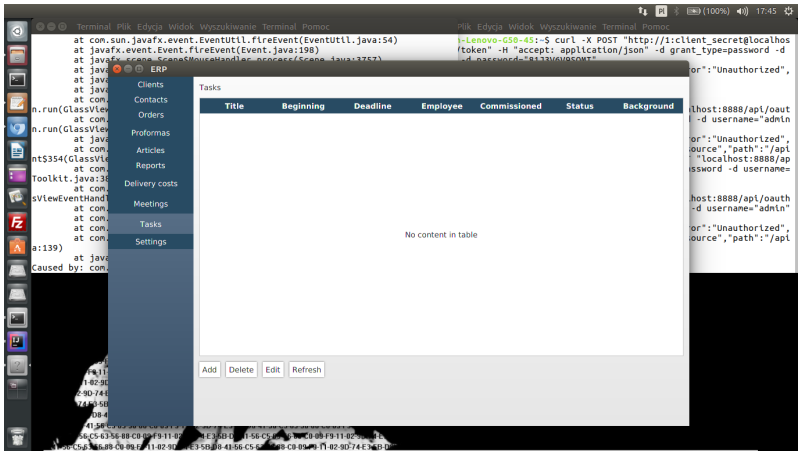
library settings



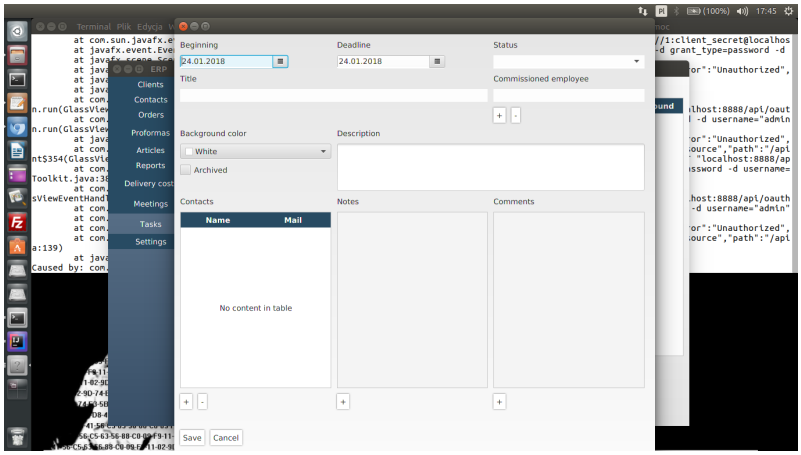
profile settings



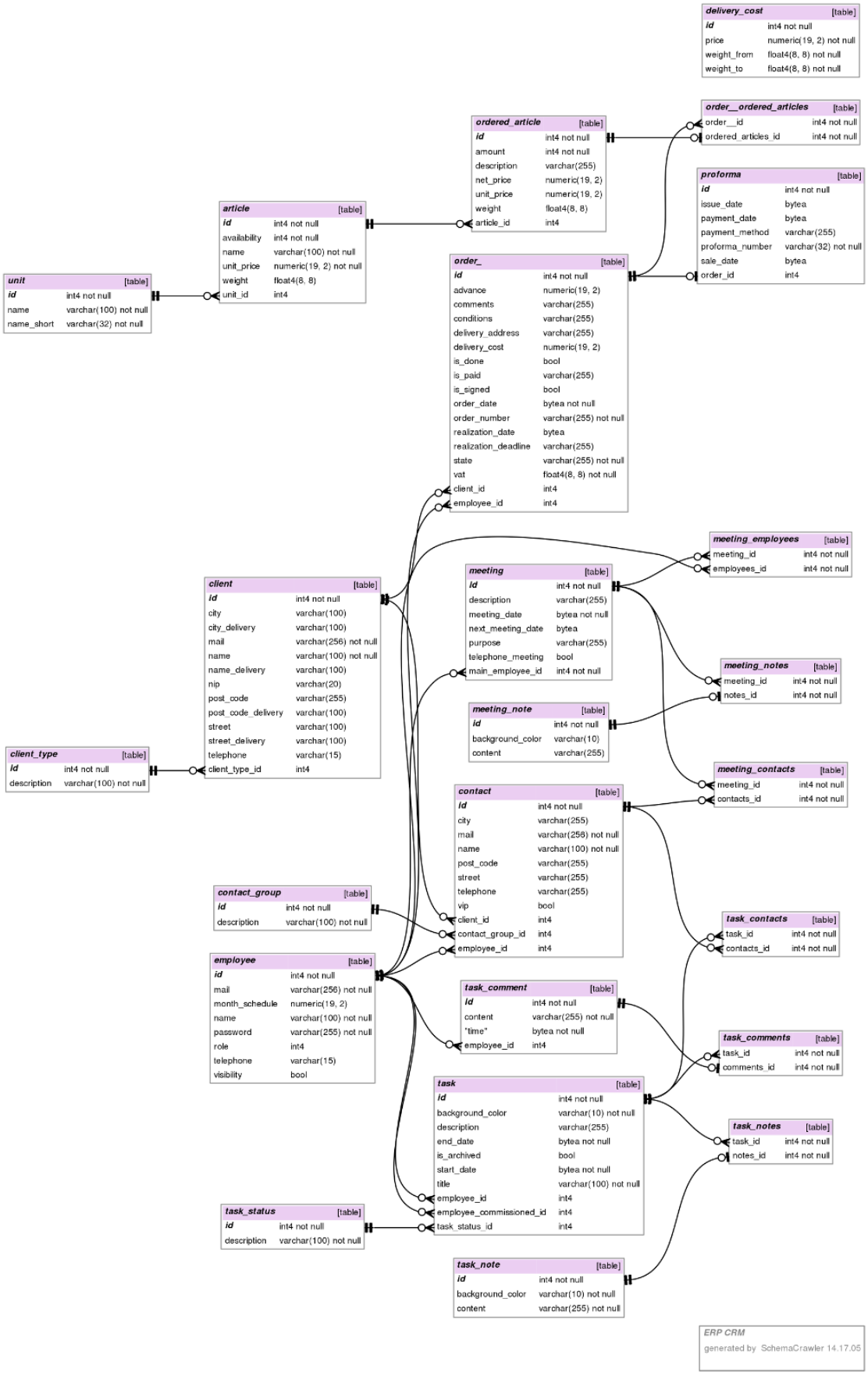
mail settings



tasks



adding tasks



Entity Relationship Diagram