MASARYK
UNIVERSITY

FACULTY OF INFORMATICS

The solution to a selected problem in the domain of open data

Master's Thesis

BC. TIMOTEJ ŠULÍK

Brno, Spring 2024

MASARYK UNIVERSITY

FACULTY OF INFORMATICS

The solution to a selected problem in the domain of open data

Master's Thesis

BC. TIMOTEJ ŠULÍK

Advisor: Ing. Leonard Walletzký, Ph.D.

Department of Computer Systems and Communications

Brno, Spring 2024


FACULTAS ARTIS INFORMATICÆ
UNIVERSITAS MASARYKIANA

Declaration

Hereby I declare that this paper is my original authorial work, which I have worked out on my own. All sources, references, and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

Bc. Timotej Šulík

Advisor: Ing. Leonard Walletzký, Ph.D.

Acknowledgements

I would like to thank Ing. Leonard Walletzký, Ph.D. and Mgr. Zuzana Schwarzová for introducing me to the field of open data, Mgr. Dominik Skála for his willingness and support with initial technical issues when reproducing his work, Operátor ICT for offering me consultations and code reviews, and last but not least, Hradec Králové for the opportunity to collaborate on the deployment of the Local Open Data Catalog (LKOD) in their municipality.

Abstract

Due to an evolution of the Czech open data cataloging standard DCAT-AP-CZ built on top of the European DCAT-AP 2.0.1, support for the popular data cataloging application CKAN in the National Open Data Catalog (NKOD) is deprecated. Municipalities and organizations using CKAN to publish open data are consequently required to move to a different local open data catalog (LKOD). One such municipality is Hradec Králové, which has selected the Local Open Data Catalog (LKOD) by Operátor ICT as the new solution, but is lacking resources to implement it. Therefore, the aim of this thesis is to carry out the implementation of LKOD in Hradec Králové.

The first part of the thesis explores the need for open data cataloging and the underlying standards, introduces the local and national open data catalog, compares the possible solutions, and discusses why the problem was selected.

In the second part, analysis and implementation is performed. To make the LKOD solution suitable, a necessary development is carried out to add missing, but crucial functionality of user and organization management. Then, maintenance is performed in CLM, an application implemented in prior work offering migration of datasets from CKAN to LKOD. LKOD is customized according to the needs of Hradec Králové, an implementation strategy is selected, and LKOD is deployed into production. To conclude the implementation, the deployed LKOD is registered in the National Open Data Catalog (NKOD).

Changes beneficial for other municipalities and organizations interested in using LKOD are submitted to the upstream repositories, and the deployment process is documented, making the outputs of this thesis a useful resource for other municipalities or organizations interested in using LKOD.

Keywords

Open data, OFN, DCAT, DCAT-AP, DCAT-AP-CZ, Open Data Catalog, LKOD, NKOD, CLM, Operátor ICT, OICT, Open Source, Hradec Králové, Deployment, SeSLab

Contents

List of Tables

List of Figures

Acronyms

CI/CD Continuous Integration and Continuous Delivery. 29, 39

CKAN Comprehensive Knowledge Archive Network. 1, 9–11, 14, 15, 17, 18, 24, 38–40, 46, 47

CLM CKAN LKOD Migrator. 1, 14, 15, 17, 18, 20, 23, 24, 38–41, 46, 47, 52, 72

CRUD Create, Read, Update, Delete. 21, 22

DCAT Data Catalog Vocabulary. 9

DCAT-AP DCAT Application Profile. 4, 5, 9, 49

DIA Digitální a informační agentura [Digital and Information Agency]. 5, 9

HDT Header, Dictionary, Triples. 5

JSON-LD JavaScript Object Notation for Linked Data. 8, 11, 12, 14, 16

JWT JSON Web Tokens. 12, 13

LKOD Lokální katalog otevřených dat [Local Open Data Catalog]. 1, 6–8, 10–17, 19, 21–23, 25–30, 32–34, 37, 38, 40, 42–46, 48–50, 52, 72

NKOD Národní katalog otevřených dat [National Open Data Catalog]. 3–9, 11, 12, 14, 33, 45, 47, 48, 52

OFN Otevřená formální norma [Open Formal Norm]. 4, 5, 9, 11, 14, 46

RDF Resource Description Framework. 5

REST Representational State Transfer. 12, 13, 15

RÚIAN Registr územní identifikace, adres a nemovitostí [Register of territorial identification, addresses and real estate]. 15

SPARQL SPARQL Protocol and RDF Query Language. 5, 6, 11, 12, 14, 16, 27, 32, 43, 47, 49

WAF Web Application Firewall. 32, 47

Introduction

The aim of this thesis is to familiarize with the current state of open data in the Czech Republic, identify a problem in the open data domain, and subsequently design and implement a solution. The entire process is to be carried out with the guidance from Operátor ICT, a company responsible for the development of an open data cataloguing solution in Prague, called the Local Open Data Catalog (LKOD).

During analysis, a primary problem is identified, being the deployment of LKOD for the municipality of Hradec Králové. Two prerequisites are also identified; a necessary development in the LKOD project, and maintenance in a related application implemented in prior work, the CKAN to LKOD Migrator (CLM), which allows migrations from a no longer supported cataloging solution CKAN.

An extensive development is carried out to address the shortcomings of LKOD and CLM, which is then reviewed and verified by Operátor ICT. Next, customization and deployment details are communicated with the Hradec Králové IT team, and the LKOD project is forked and modified in accordance with the agreed upon description. For evaluation before deployment into production, the project is deployed onto the infrastructure of the Laboratory of Service Systems (SeSLab), where it is verified by the Hradec Králové IT team. After confirmation, the project is deployed into production in Hradec Králové, tested, and finally registered in the National Open Data Catalog (NKOD).

The thesis is divided into eight chapters. The first chapter discusses the concepts and relevant legislature concerning open data in the Czech Republic. The second chapter analyzes the existing technical solutions maintained by Operátor ICT. In the third chapter, the process of identifying and choosing a problem to solve is discussed, and justification is given for why the problems were deemed the most important and addressed. The fourth chapter performs analysis of the work that needed to be carried out to address the identified problems, sets proper expectations, and lists methods of verifying the produced results. In the fifth chapter, the implementation is executed, and the technical aspects of the changes are discussed. The sixth chapter discusses unforeseen challenges that have arisen during implementation. Then follows the seventh chapter, where a number of tasks are listed

as a part of future work. Finally, the work is concluded in the eight chapter, summarizing the result of the efforts.

1 Open data in the Czech Republic

Open data has become a crucial part of the modern information environment, especially in the public sector. This trend is captured in legislative measures on Free Access to Information, which defines open data as information published in an open and machine-readable format enabling remote access. [1, 2, 3] This data is an important source for the creation of new services and applications that enhance the efficiency of public administration and increase transparency. [4]

Open data in the Czech Republic is not just a theoretical concept; it creates tangible benefits for citizens and organizations. It is important to realize that open data is not limited to the area of transparency in public administration. [5] It is relevant to a wide range of fields and can provide a foundation for innovative solutions and services across various sectors, both in non-commercial and commercial applications. [2] Projects such as Government Watchdog [1], Maps without Barriers[2], Where to take it? [3], or the expansion of Seznam.cz's map of Prague with public transport route search are examples that illustrate the practical use of open data to improve citizens' lives. [4]

Other applications utilizing open data solve problems ranging from improving public transportation in other cities, discovering science and technology, making an informed decision when selecting secondary education, to entertaining ones like building a realistic copy of the city Brno in the video game Minecraft. [6]

1.1 Open data requirements

Successful utilization of open data requires meeting certain conditions. Open data must not only be accessible in a machine-readable format but also be accompanied by usage conditions, recorded in the National Open Data Catalog (NKOD), and supported by documentation. Additionally, it should be easily accessible without technical barriers for example the need to create an account on a website. [4, 7] The primary

[1] Tracks whether politicians and officials are not misusing funds.
[2] Helps people with reduced mobility to better plan trips.
[3] Search for places where you can legally dispose unwanted things.

specification capturing the relevant technical details is Open Formal Norms (OFN)[4], which is the basis for successful cataloging of open data in the Czech Republic.

While the legislation mandates the publication of open data in specific scenarios, entities are not restricted to only these situations; they have the option to voluntarily release additional open data. This approach is supported within the mentioned law, which allows the publication of information without a legal obligation, provided it does not violate the law. Importantly, such information should be provided in accordance with open formats and technical specifications to ensure easy use of the data. [1, 4]

1.2 Cataloging of open data

A fundamental characteristic of open data is its easy accessibility to the public. This necessitates the use of a catalog, which captures metadata about published data in a well-defined, mandated, and documented form. [8] An important fact about open data catalogs is that a catalog primarily stores metadata about the published data, while the actual data described by the entries in an open data catalog are stored and provided by each data provider according to their preference. [9] It is crucial and mandatory for all data providers to have their data cataloged in at least one data catalog, the National Open Data Catalog (NKOD). [8]

1.2.1 DCAT-AP, OFN and DCAT-AP-CZ

The DCAT Application Profile for data portals in Europe (DCAT-AP) is a specification aimed at improving the accessibility and interoperability of public sector datasets[5] across Europe. [10]

Open Formal Norms (OFN) refers to a Czech standard defining the ways of organizing and sharing of open data. These norms provide technical recommendations focused on selected datasets to ensure

[4] `https://data.gov.cz/ofn/`

[5] A dataset is a set of related data made available through one or more distributions.

interoperability, enabling easier utilization of such data regardless of the provider. [11]

DCAT-AP-CZ refers to a specific OFN which defines the ways in which data should be exposed in order to be harvestable by a National Open Data Catalog (NKOD). It starts with defining the data model for cataloging records and then outlines two ways to harvest data, based on the DCAT Application Profile (DCAT-AP) 2.0.1 standard - using a SPARQL endpoint or using data files. Both approaches are discussed in detail in the sections 1.2.2 and 1.2.3. DCAT-AP-CZ ensures consistency in describing datasets and their distributions, regardless of how the data is represented. It distinguishes between datasets, their distributions, and data services. Distributions can either be downloadable files or accessed through data services, with the aim of being equivalent in content. [12]

Ultimately, DCAT-AP, OFN, and DCAT-AP-CZ aim to achieve the same goal: enhance data accessibility and ensure machine readability. These frameworks and standards are designed to facilitate the sharing and utilization of public sector datasets by providing guidelines for organizing, describing, and sharing data in a standardized manner. By adhering to these specifications, data publishers can ensure that their datasets are easily discoverable, understandable, and interoperable across different platforms and systems.

1.2.2 The National Open Data Catalog

It is mandatory for every Data Provider to catalogize the published data in the National Open Data Catalog (NKOD), operated by the Digital and Information Agency (DIA). NKOD is a publicly available platform, where data from all data providers is aggregated and displayed in an organized manner in a web interface. For consumption by applications, the aggregated data is exposed in numerous machine-readable formats, including RDF TriG, HDT, JSON, CSV, and exposed using SPARQL, GraphQL, and Linked Data Fragments endpoints. [13, 14]

NKOD offers two distinctive ways in which data providers can utilize to publish their data.

Firstly, a data provider can populate NKOD manually, which is a convenient way for organizations that have limited technical capa-

bilities or staffing, and are planning to publish only a small number of datasets. [15] This solution is officially recommended by the open data portal for organizations that are planning to share less than ten datasets. The user manually fills in a dataset creation form, and submits a generated document manually to a designated Data Box[6] (datová schránka). [8]

Secondly, a data provider may maintain their own local open data catalog, in which they catalog their datasets. If the local catalog meets the technical parameters described in the relevant standards, the provider is not required to manually transfer records to NKOD, but it is possible to ensure automatic synchronization of the content of the local catalog with NKOD. [8] This approach is discussed further in the following chapter.

The focus in this thesis is placed onto the second solution, where NKOD is populated by data from a local catalog automatically.

1.2.3 The local open data catalog

A local open data catalog (LKOD) is an application, which can optionally be hosted by an obliged entity using their own means. LKOD plays a similar role to NKOD, meaning it too stores and aggregates metadata about datasets and their distributions, but on a local scale.

The term local open data catalog does not refer to a single implementation of an application, rather to a set of rules and an interface which need to be followed in order for an application to act as a local catalog. Most importantly, the mandatory data of the catalog needs to be accessible in a well-defined standard format. At the time of writing, NKOD supports three ways in which data can be published for NKOD harvestation - the CKAN API (legacy), DCAT-AP Documents, and DCAT-AP SPARQL Endpoint. Each of these solutions is discussed in the following chapter.

The main benefit of hosting a local open data catalog is simplification and automatization of dataset publishing into NKOD, at the cost of a higher technical complexity. When an LKOD is in use, additions, updates, and removals of datasets are no longer submitted manually into NKOD using a Data Box one at a time. Instead, they are entered

[6] An electronic repository for the delivery of documents to public authorities.

locally into LKOD, from where they are harvested automatically by NKOD, without the need to notify any government entity about any updates. [8]

2 Existing solutions

2.1 LKOD applications

A local open data catalog is an abstract term, and it has been materialized in many different forms, with various levels of complexity. A referential implementation is also provided, which will be discussed first, then followed by two, more complex solutions.

2.1.1 LKOD referential implementation using GitHub

The first way of maintaining a local catalog is demonstrated in a simplistic reference implementation, which uses a GitHub repository for the storage of the dataset data, and GitHub actions for automatic generation of a catalog file, which is then consumed by NKOD. The entry of dataset information is performed using an official dataset form[1] provided by `data.gov.cz`, and the resulting data is then downloaded in the JSON-LD format, to be committed into the git repository.

This approach is free, and requires no custom infrastructure; it is also fully supported, meaning all available attributes can be provided, making the resulting data exported to NKOD equivalent to what could be provided if a more complex LKOD solution was used. On the other hand, the dataset files are managed manually, which means the process is potentially error-prone, and it requires nontrivial technical knowledge, primarily regarding the usage of git and GitHub. Another disadvantage is that no public facing catalog user interface is provided with this approach, and the data is displayed in a presentable way for public consumption only in NKOD.

A minimalistic repository that has been prepared to be forked is available at `https://github.com/datagov-cz/lkod-min`, with a readme file offering concise instructions. Out of the available ways of NKOD connection, this implementation uses *DCAT-AP Documents*.

[1] `https://data.gov.cz/formulář/registrace-datové-sady`

2.1.2 CKAN

The Comprehensive Knowledge Archive Network (CKAN) is an open source data management system, which is seeing wide use by various organizations throughout the world, both governmental and non-governmental. It operates with three entity types, Dataset, Organization and User. CKAN includes the option to store the actual dataset files withing itself, where they are referred to as resources. [16] The datasets and the related information can be edited in an intuitive user interface, and accessed via a HTTP API, which uses a structure that has been designed and is specific to CKAN. [17]

The stored metadata in a base installation of CKAN is rather limited, consisting of only base attributes like the title, publisher, date, file formats, and license. CKAN offers the possibility to add any number of custom fields from within the base user interface in a key-value manner; however, such functionality does not offer any kind of validation, and these attributes are only displayed at the bottom of the Additional Info section. [18] The process of a dataset creation in CKAN can be found in Figure 2.1.

In addition to the default limited way of additional metadata entry, CKAN can also be enriched using extensions. Extensions allow the base set of fields to be expanded, with the possibility of implementing proper validation and first-class rendering by extending the user interface templates. [19]

Since the OFN mandated for the Czech Republic are based on DCAT-AP, an officially provided DCAT extension[2] for CKAN could, in theory, be a way of adding the necessary support to CKAN. However, it was determined by DIA (Digitální a informační agentura) that due to the extensive additions of DCAT-AP and OFN on top of DCAT, support even using such plugins is not viable. [20, 21]

The usage of CKAN along with NKOD is currently possible using the *CKAN API* publishing option, and still in use by some municipalities; however it has been marked as deprecated, and its usage is discouraged[3], as data provided by CKAN is severely lacking.

[2] `https://extensions.ckan.org/extension/dcat/`

[3] `https://ofn.gov.cz/rozhraní-katalogů-otevřených-dat/2021-01-11/#ckan-api`

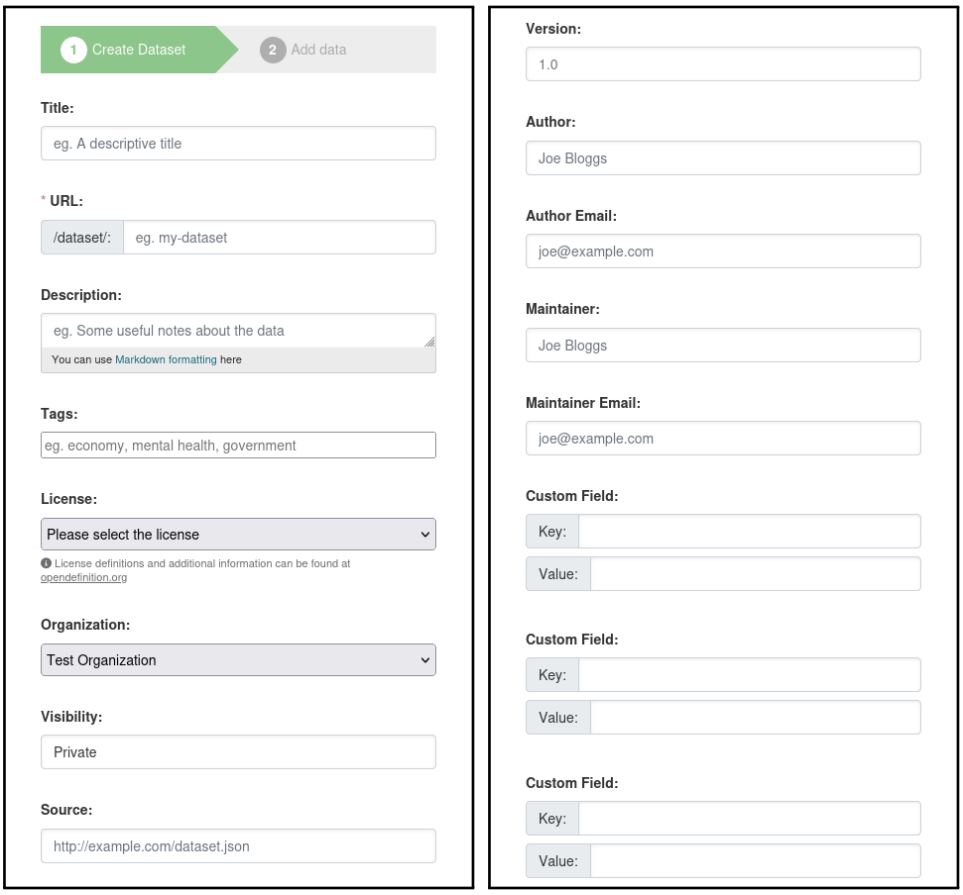

Figure 2.1: Create Dataset form in CKAN 2.10. Source: Author.

2.1.3 LKOD by Operátor ICT

LKOD by Operátor ICT is a complete LKOD solution offering an administration user interface, and fully separate a public-facing catalog. Being similar to CKAN, it offers functionality allowing the users intuitive management of datasets and their distributions, and it also offers integration with an FTP server in order for users to have the ability to upload files from the administration user interface. LKOD has been developed as a response of Prague to CKAN no longer being a sufficient storage of the relevant information, the reasons for which being discussed in the previous chapter. The user interface of the public catalog is depicted in Figure 2.2.

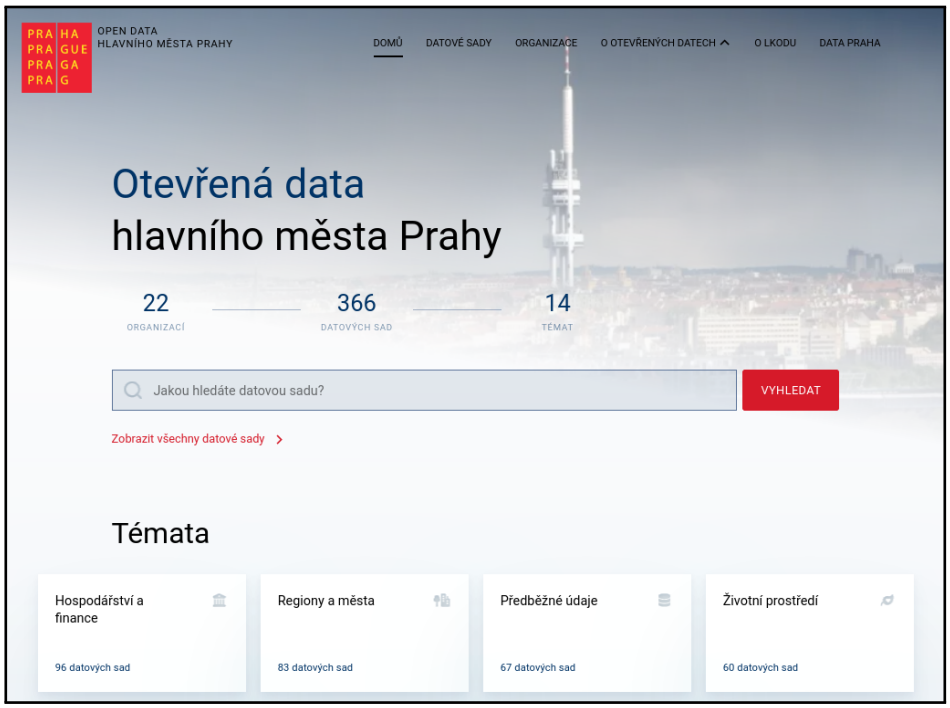

Figure 2.2: LKOD Catalog at opendata.praha.eu. Source: Author.

Unlike CKAN, LKOD by Operátor ICT does not handle the entry of dataset and distribution metadata on its own, but it delegates the task to the official web interface provided by `data.gov.cz`. The data entered in the official form is then returned back to LKOD, which stores it both in a PostgreSQL database and a SPARQL server, and exposes it both using a JSON-LD endpoint and a SPARQL endpoint. Therefore the NKOD publishing method can be either *DCAT-AP Documents* or *DCAT-AP SPARQL Endpoint*.

Due to the generality of such approach, meaning the dataset metadata is provided by `data.gov.cz` directly, this solution is future-proof. Any potential updates to OFN can be easily incorporated into the solution without the need for major architectural changes or customizations, as opposed to e.g. CKAN.

LKOD is built using open-source components, and it is itself, open sourced under the MIT License, allowing anyone to freely re-use the source code. From a technical perspective, LKOD is divided into seven

distinct components, which interact with each other to form the complete LKOD solution.

As the rest of the thesis is focusing specifically on the LKOD implementation from Operátor ICT, from now on, LKOD is going to refer to this specific implementation, rather than being used as a generic term.

Now follows discussion of each component of the LKOD solution.

Database

The database of choice for the LKOD project is PostgreSQL. At the time of writing, the used distribution and version is from Docker Hub, specifically `bitnami/postgresql:13.9.0`.

Database migrations are performed using the `db-migrate` library. The details of migrations are discussed in the Backend section, as the migrations themselves are a part of the backend project.

The database consists of 9 tables, 3 of which are used for secondary uses like password resets, activity logging, and migration tracking.

Backend

The LKOD backend (internally named `lkod-backend`) is responsible for the management of datasets, storage of the entered information in a SPARQL database, and since recently, also automatic dataset synchronization from ArcGIS. Data for the two frontends is exposed using a combination of REST endpoints and a SPARQL endpoint. Data for NKOD is exposed in the JSON-LD format. The REST endpoints are authenticated using JWT, and documented in the OpenAPI 3 format. The backend project includes database migrations utilizing the `db-migrate` NodeJS DB migration framework, where the migrations are checked and possibly executed when the backend Docker container starts.

The backend is written in TypeScript with the use of the Express.js Node.js application framework. Testing is performed with the help of the *mocha*, *sinon*, *supertest* and *chai* libraries.

Frontend: Catalog

The LKOD catalog (internally named `lkod-catalog`), being the public-facing part of the solution, displays all data and metadata about organizations, datasets, and topics to the general public. The datasets and organizations are searchable using simple text search. The data is fetched from the backend using SPARQL, and the majority of the content is rendered on the server using NextJS server-side rendering (SSR).

The catalog is written in TypeScript in the NextJS 14 framework, and it is not automatically tested.

Frontend: Administration

The second frontend, LKOD administration (internally named `lkod-admin`), is separate from the public-facing LKOD catalog.

The administration interface is used only by the parties responsible for the published data, so it is separate from the public-facing catalog frontend. The original functionality only includes the creation, update, and deletion of datasets and their distributions. There is no organization or user management, meaning such data has to be inserted into the database manually. This frontend interacts with the backend using REST with JWT, and the majority of the pages are rendered using NextJS server-side rendering (SSR).

The administration frontend utilizes the exact same tech stack as the LKOD catalog, being TypeScript & NextJS 14. This frontend is also not tested.

Redis

Redis is present, at the time of writing, the used distribution being `bitnami/redis:6.2` from Docker Hub.

The only use of Redis is for a blacklist of JWT tokens for the backend, where each JWT token is blacklisted during logout.

SPARQL storage

For the SPARQL database, Apache Jena Fuseki 2 is used. A custom fork[4] is used, forked around December 03, 2022, but the only change is the addition of a Dockerfile and a GitLab CI build.

Besides PostgreSQL, the catalog data is also kept inside the SPARQL database; however, as previously mentioned, only the LKOD catalog makes use of the SPARQL endpoints. NKOD is populated using JSON-LD instead.

Optional FTP storage

The FTP storage part of the stack is optional because the LKOD is merely a catalog, and hosting of the distribution files themselves are not strictly a feature of the catalog. However, due to the familiarity of users with CKAN, where the storage is integrated into the catalog, LKOD also offers integration with an FTP and NGINX server, where the files can be uploaded from the LKOD administration user interface itself.

After the file is uploaded using FTP from the LKOD backend, it is made public using an NGINX instance configured to expose files from the mounted directory. The generated public link can then be copied from the LKOD administration user interface, and included into the dataset as a distribution.

For the FTP storage part of the stack, a combination of Pure-FTPd and NGINX are used, the versions being `crazymax/pure-FTPd:1.0.49` and `bitnami/nginx:1.22` at the time of writing.

2.2 CLM

Before the evolution of OFN started demanding a more elaborate LKOD solution, CKAN has been the platform of choice for a number of municipalities and organizations. After the evolution of OFN, there arose a need for a way to efficiently migrate datasets from CKAN to LKOD. This need lead to the development of the the CKAN LKOD Migrator (CLM) by Mgr. Dominik Skála, which allowed such migra-

[4] `https://gitlab.com/operator-ict/golemio/extras/fuseki2`

tion in an intuitive user interface without the need to transfer datasets between platforms manually. [22]

The CLM project has been adopted by Operátor ICT, and the application is currently publicly accessible at migrace.opendata.praha.eu, allowing any municipality or organization to use it without the need to deploy it on their own infrastructure.

This migration performed by CLM is achieved by accessing the CKAN datasets with the help of a CKAN API key, and inserting them into LKOD using its REST interface, after authenticating using a username and a password. [22]

Additional features of the application are the possibility of prefilling selected properties for each imported dataset, namely the publishing frequency, the Register of territorial identification, addresses and real estate (RÚIAN) code, and the license of the distribution. [22]

CLM is implemented in Python, with the help of the Flask framework, and additional libraries `flask_bootstrap`, `flask_wtf`, `wtforms` and `jsonschema`. Small parts of the user interface are managed by jQuery 3.5.1.

3 Selecting a problem to solve

After delving into the open data field, the next step was to learn about current pain points, learn about the priorities from the relevant sources, and then identify & address selected issues. The main sources of information were Operátor ICT, the company responsible for open data in Prague, and prior work from Mgr. Dominik Skála, a former member of the Laboratory of Service Systems, who has been solving such problems in the past. Communication was initially carried out via e-mail, and later via a Slack channel belonging to Operátor ICT. Now follows discussion of the identified issues, and the selection process.

3.1 From Operátor ICT

The primary source of possible improvements in the field of open data, and their respective relevancy, is Operátor ICT themselves. In mutual discussions with Operátor ICT, a number of tasks were proposed and discussed.

3.1.1 Populate LKOD Catalog using JSON-LD

Currently, the code driving the LKOD Catalog user interface performs SPARQL queries and renders the fetched data on the server side, later passing it to the frontend side in a pre-rendered manner. Due to the identified high cost of SPARQL operation[1], this process could be optimized by populating the user interface using existing JSON-LD endpoints instead. The replacement of SPARQL with JSON would also fully remove the dependency on the SPARQL server, which could then be disabled completely, making it an optional component of the LKOD solution.

As the immediate benefits are questionable for municipalities other than Prague, it was decided to let this issue be handled by the developers of Operátor ICT.

[1] `https://gitlab.com/operator-ict/golemio/lkod/lkod-general/-/issues/106`

3.1.2 Features for Datové centrum Ústeckého kraje

Datové centrum Ústeckého kraje (Data center of the Ústí Region) have expressed interest in having the possibility to manage organizations within the LKOD administration user interface.

Due to an equal request also coming from another party, discussed in the next section, the addition of organization management is one of the improvements implemented as a part of this thesis.

3.1.3 Implementation of LKOD Hradec Králové

Hradec Králové has contacted Operátor ICT with the interest of having LKOD implemented in their municipality, asking for technical cooperation. Due to the lack of resources on the side of Operátor ICT, this task was not going to be put in planning in the foreseeable future. As a result, implementation of LKOD in Hradec Králové has been chosen to be performed as a part of this thesis.

The implementation of LKOD in Hradec Králové not only necessitates the deployment of LKOD itself, but also customization of the user interface according to the Hradec Králové visual style, and also the implementation of user and organization management in the LKOD administration user interface.

Due to the presence of CKAN in Hradec Králové, the use of CLM is assumed, therefore evaluation and possible maintenance of CLM is to be performed.

3.2 From prior work

In the future work part of his thesis, Mgr. Dominik Skála has identified a set of potentially beneficial improvements both for his application CLM, and for LKOD itself. [22] The individual improvements are discussed below.

3.2.1 For CLM

Transformation of distributions of datasets into datasets

In prior work, it was identified that in some instances, municipalities tend to add multiple unrelated, distributions into a single dataset. [22] The idea is to implement a feature where these cases could be handled intuitively in the migration application itself.

Although a potentially useful feature, such improvement was not selected for this thesis due to the lack of interest of the questioned parties.

Default settings for individual datasets

In the current state of CLM, it is possible to pre-fill various properties like periodicity, license, and the RÚIAN value during the migration for each imported dataset. The idea here is to offer an extension of the user interface where such information could be pre-filled for each dataset in the CLM user interface. [22]

This potential improvement has not been implemented due to the fact that it will likely be necessary to go through each dataset anyway, and correcting such properties is fairly straight forward due to the intuitiveness of the dataset modification user interface. The majority of datasets of Hradec Králové only contain a small number of distributions, hence the limited usefulness of such feature. Also due to the lack of interest, focus is better spent elsewhere.

Migration from applications other than CKAN

The core of this feature is the possibility to migrate from other applications, for example DKAN[2]. [22]

Identifying or contacting any specific municipality showing interest in such feature has not been successful. Due to the lack of interest, the decision is to not implement support for additional applications in CLM as a part of this thesis.

[2]DKAN is an open-source open-data platform inspired by CKAN built on top of the Drupal CMS (Content Management System). [23]

3.2.2 For LKOD

The possibility of adding individual organizations via the GUI

The goal, being identical to aforementioned request from Datové centrum Ústeckého kraje presented in the section 3.1.2, is to extend the LKOD administration with the possibility of managing organizations from within the web user interface, removing the need to perform raw SQL queries when carrying out such a task. [22]

Due to this feature being in demand, after a consultation with Hradec Králové, who have also expressed interest, it was identified as one of the priorities and consequentially implemented. The analysis and implementation are discussed in subsequent chapters.

Installation process via web application

Currently, the installation of LKOD requires a person with nontrivial technical abilities. The idea for this improvement would be to create an install wizard, offering to enter the initial user and organization information in a web interface. [22]

LKOD is a fully custom web application consisting of multiple technical components, and its deployment will always require a high amount of technical knowledge. When a technician is able to perform the advanced tasks needed for the deployment of LKOD, they will likely also be able to execute a single SQL command needed to insert an initial user into the database. After the implementation of user & organization management from within the web user interface, this initial user is all that is needed to have full control over all data needed for LKOD operation, including adding more users and organizations. The conclusion is that the effort is best spent at optimizing the day-to-day operation by eliminating the need to ever perform raw SQL queries after the installation process.

3.3 Conclusion

Due to the lack of resources on the side of Hradec Králové and Operátor ICT, and due to the general usefulness of the solution for the Hradec Králové region, implementing LKOD in Hradec Králové has

been selected as the main objective of this thesis. Due to the supposed use of CLM in Hradec Králové, maintenance of the CLM application is to be performed. Lastly, to eliminate the need to perform SQL queries when operating the deployed solution, a user & organization management interface will be implemented.

The scope of the aforementioned modifications is discussed in the following chapter, including acceptance criteria and a testing strategy.

4 Analysis

In the previous chapter, a set of the most beneficial objectives was identified, along with the justification for each of them. Now follows discussion of the objectives and their acceptance criteria.

4.1 LKOD administration user interface

4.1.1 User management

Currently, it is impossible to perform basic Create, Read, Update, Delete (CRUD) operations on rows of the `users` database table from within the LKOD administration user interface. Consequentially, every request to update the `users` table has to be handled by a skilled sysadmin. Such a need removes from the intuitiveness of the application, and it might discourage municipalities from giving access to more users, for example from other organizations. This can, in turn, lead to the loss of potentially publishable data due to high barrier to entry.

Therefore, user management should be possible from the LKOD administration user interface. User management needs to be accessible only to a special subsets of users, designated using the later discussed `superadmin` role.

Now follows a list of acceptance criteria; a logged in user with the new `superadmin` role is assumed.

Acceptance criteria

UM1: It is possible to create, read, update, and delete users.

UM2: Before a user is deleted, a confirmation prompt requiring the name of the user to be typed in is displayed.

UM3: Deleting a user cascade-deletes all datasets that they are the owner of.

UM4: The current user can not delete nor modify their own account.

4.1.2 Organization management

Currently, it is impossible to perform CRUD operations on rows of the `organizations` database table from within the LKOD administration user interface. Similarly to the management of users, such an operation is not intuitive, and it might discourage the owners of LKOD from adding other organizations, potentially losing valuable publishable data.

It needs to be possible to manage organizations from the LKOD administration user interface. Organization management needs to be accessible only to a special subsets of users, designated using the later discussed `superadmin` role.

Now follows a list of acceptance criteria; a logged in user with the new `superadmin` role is assumed.

Acceptance criteria

OM1: It is possible to create, read, update, and delete organizations.

OM2: Before an organization is deleted, a confirmation prompt requiring the name of the organization to be typed in is displayed.

OM3: Deleting an organization cascade-deletes all datasets that belong to it.

OM4: It is possible to read, add, and remove members of an organization.

4.1.3 Superadmin role

Currently, a user in the LKOD system can have one of two roles, `user` or `admin`. The role `admin`, having a misleading name, does not offer any real "administrative" capabilities. Users carrying the role `admin` have the ability to modify all datasets from their organization, in contrast to the role `user`, the carriers of which can only modify datasets that they have created.

This brings the need for a third role, which would enable its carrier to have even higher privileges than `admin`. The elevated privileges would be to perform user and organization management, and have edit access for datasets belonging to every organization in the system.

After a discussion with Operátor ICT, the newly added role would be named `superadmin`.

Acceptance criteria

SA1: The column `role` in the database table `users` accepts a third enumeration constant named `superadmin`.

SA2: A migration modifying the database column is added.

SA3: A user with the role `superadmin` can log into the LKOD administration user interface.

SA4: A logged in user with the role `superadmin` sees the *Organizations* and *Users* bookmarks in at top of the LKOD administration user interface.

SA5: A logged in user with the role `superadmin` can create, read, update, and delete all organizations.

SA6: A logged in user with the role `superadmin` can create, read, update, and delete all users.

SA7: A logged in user with the role `superadmin` can create, read, update, and delete datasets of all organizations.

SA8: A logged in user with the role `superadmin` can re-assign the owner of each dataset.

SA9: A logged in user with the role `superadmin` can perform all actions `admin` already can.

SA10: The existing roles `user` and `admin` stay unchanged, not having any more permissions than described in the `docs/user_roles.md` document.

4.2 CLM application maintenance

Since its initial deployment, the CLM application has had only routine maintenance done on it, for example minor updates for compatibility with update data schemas. After testing the application, a set of necessary modifications and updates was identified; discussed below.

4.2.1 CKAN access token is too long

As a part of ongoing development of CKAN, the length of the generated access token required for CLM to function has increased in length. The access token no longer fits into the designated field in CLM, rendering the application unusable with recent versions of CKAN.

Acceptance criteria

CLM-TK1: The CLM application should accept CKAN access token at least 256 characters long.

4.2.2 Internal Server Error when loading datasets

When built from source, the application crashes with an Internal Server Error when loading datasets from CKAN. The issue does not manifest on the currently deployed version.

Acceptance criteria

CLM-CR1: The CLM application should not fail with an Internal Server Error when loading data to migrate.

4.2.3 Data leak in the CLM user interface

The CLM application manifests confusing behavior where revealing the dataset error dialog in the user interface leads to duplication of errors from the previous times the dialog was opened.

Acceptance criteria

CLM-CF1: When opening the dialog with dataset conversion errors multiple times, the errors should no longer duplicate each time the dialog is opened.

4.2.4 Migrate HTTP distribution links to HTTPS

During migration, the CLM application uses the original HTTP links for dataset distributions hosted outside of CKAN. However, in some

cases, example being Hradec Králové, such links could be automatically upgrade to HTTPS, which is required by the best practices for open data. [24]

Acceptance criteria

CLM-HTTPS1: When converting a distribution, an external distribution file HTTP link should be migrated to HTTPS automatically as long as the file succeeds to be accessed with HTTPS.

4.3 LKOD implementation in Hradec Králové

A successful deployment of LKOD in Hradec Králové necessitated thorough planning, which was performed in conjunction with the Hradec Králové IT team. It was identified that in order for the deployment to be suitable for production, LKOD needs to be modified with the relevant visual style of the municipality, deployed into the infrastructure according to the best practices, tested, and the relevant staff will need to be trained to allow unguided LKOD operation in the future. The aforementioned best practices have been discussed both with the technician responsible for the infrastructure, and with a contractor providing and maintaining technical equipment for the municipality. The acceptance criteria for all of the requirements are discussed below.

4.3.1 Modification

Acceptance criteria

HK-M1: The default Prague branding of LKOD is fully replaced by branding of Hradec Králové, according to the Hradec Králové visual style.

HK-M2: The ”DATA PRAHA” bookmark in the LKOD Catalog is replaced by ”HRADEC V DATECH”, leading to the URL `https://hradecvdatech.cz/` instead.

HK-M3: Text on all pages of the LKOD Catalog are replaced by texts prepared by the Hradec Králové IT team.

HK-M4: The social network links on the bottom of the site are replaced with relevant Hradec Králové social links.

4.3.2 Deployment

A decision was made by the Hradec Králové IT team that existing on-premise infrastructure would be used for the deployment of LKOD. With this decision in mind, the following strategy and deployment parameters were selected.

System requirements

An informed decision is supported by data, where it was reported by Operátor ICT that LKOD is visited by around 1,000 people, translating to around 4,000 pageviews per month. With this information in mind, in a subsequent discussion a developer from Operátor ICT, it was identified that the optimal system requirements with a substantial headroom for the deployment of LKOD in Hradec Králové are as follows:

- 4 CPU cores
- 8 GB of RAM
- A Debian-based OS
- At least 32GB of disk storage

The CPU, RAM, and disk space can be grown on demand by adjusting the parameters of the virtual machine, also known as vertical scaling.

Domain selection

The Fully qualified domain name (FQDN) to be used for LKOD was decided to be `opendata.mmhk.cz`. The LKOD application uses a main domain for the catalog, and then the following subdomains for the respective parts of the LKOD application:

- `api`: The LKOD backend
- `admin`: The LKOD administration frontend
- `sparql`: The SPARQL endpoint
- `ftp`: Public files uploaded onto the FTP

HTTPS configuration

HTTPS in Hradec Králové for all web solutions is handled in a central place, the Web Application Firewall (WAF), which is maintained by an external contractor. The LKOD solution therefore only needs to expose HTTP traffic, which is then secured by the WAF on a higher level. Requests to make the relevant changes in the WAF are submitted and executed by the contractor, so the configuration of HTTPS is outside of this work.

Platform selection

The first decision is to select between a bare metal deployment onto dedicated hardware, or a virtual machine.

Bare metal Deployment onto bare metal would necessitate a purchase of a dedicated machine and installing LKOD directly onto it. Due to existing infrastructure in Hradec Králové being already set up to handle virtual machines, and numerous advantages of virtual machines like the ease of virtual scaling, backups, resiliency, and many more, it was decided against bare metal.

Virtual Machine Deployment onto a virtual machine is the better option due to the many advantages discussed previously. The Hradec Králové IT team were considering the use of a virtual machine as the only viable option, therefore it was ultimately chosen to deploy LKOD into a virtual machine running on Hradec Králové infrastructure.

Containerization solution selection

After selecting the appropriate deployment platform, the next step in the deployment strategy is to select to what degree the application will be containerized[1].

None The first option was to use no containerization at all, and install LKOD by copying the application files to the virtual machine, and then installing all dependencies. This process would have to be repeated on every update, introducing space for potential mistakes and inconsistencies. The described problem has been solved by containerization.

Due to best practices from recent years, the presence of a Dockerfile in each project, and also a working Dockerized example, not using Docker[2] would have been a lost opportunity. Containerization guarantees portability and flexibility across different different environments, no matter if development, testing, or production. [26]

The decision to use Docker was supported by a developer from Operátor ICT, who specifically warned against deployment without Docker, due to unnecessary complexity and straying away from the best practice of LKOD deployment itself, which could be hurtful in the long run, for example during future LKOD upgrades.

Docker It is possible to start Docker containers manually using the Docker command line interface. However, all parameters of containers like networking and volumes need to be specified on the command line, making restarts and future upgrades challenging. For this reason, it was decided against a standalone Docker approach, since the next option, Docker Compose, allows all of such configuration to be captured in a standardized format.

Docker Compose The first viable containerization option was using Docker Compose. Compared to the previous option, all configuration

[1] Containerization is a software deployment process that bundles an application's code with all the files and libraries it needs to run on any infrastructure. [25]

[2] Docker: A popular industry standard application containerization solution `https://www.docker.com/`

specific to the machine can be captured in a `docker-compose.yml` file, for example static environment variables, networking mode, mutual dependencies of containers, volumes, and internal networks. [27] Any potentially changing variables can be imported as environment variables from a `.env` file, which can be, for example, generated by CI/CD. [28]

Docker Compose does not offer any scaling and resilience features besides container auto-restart on crash and after virtual machine reboot, and it offers only vertical scaling by increasing the amount of resources allocated to the virtual machine it is running on.

Docker Swarm Docker Swarm, being a middle ground between Docker Compose and Kubernetes, could be the solution to potential scaling needs as it is one of the simpler orchestrating solutions. It carries the possibility to scale containers across multiple nodes with a relatively low complexity compared to Kubernetes. It also includes load balancing. However, it lacks auto-scaling features, so the containers still need to be spun up and down by hand. [29]

If scaling, high fault-tolerance and resilience of the application turn out to be necessary in the future for LKOD in Hradec Králové, Docker Swarm could act as a stop-gap solution before implementing Kubernetes.

Kubernetes Kubernetes[3] , being a fully-featured container orchestration platform, would handle virtually all challenges that would arise if the application required horizontal scaling, high availability, self-healing, service discovery, load balancing, rolling updates, and more. [29]

However, as distributed systems usually do, these features come at a cost of high complexity. The deployment and maintenance of a Kubernetes cluster requires a high amount of experience, usually requiring an employee specifically dedicated to maintenance of such solution with a job title such as Cloud Infrastructure Engineer. [29]

One advantage of using Kubernetes for LKOD deployment would be the fact that Operátor ICT use Kubernetes for LKOD deployment

[3]Kubernetes: A popular container orchestration system `https://kubernetes.io/`

in Prague themselves. However, after a discussion with a developer from Operátor ICT, it was discovered that the reason for LKOD in Prague using Kubernetes was that they already had a Kubernetes cluster set up running many more applications, and it was simply easier to re-use their existing scripts and CI configurations for the deployment of LKOD. The developer from Operátor ICT stated that for deployment of LKOD on its own, Kubernetes would likely provide little value in the case of Hradec Králové, and Docker Compose would be the optimal solution due to the lack of need for scaling, and the simple nature of the application.

Conclusion Due to the low expected page visit counts for Hradec Králové, there is no predicted need for horizontal scaling, making features like automatic scaling, load balancing, and high availability of no use. The presence of these features offered by each of the containerization solutions are compared in the table 4.1.

With the support from Operátor ICT, it was ultimately decided that the Docker Compose approach would be ideal due to its low complexity, compared to more advanced solutions like Kubernetes. Even though Docker Compose is usually used only for development and testing, for simple single node deployments, it is a perfectly valid and officially supported option [30]. Therefore, Docker Compose was chosen as the optimal solution and used.

Table 4.1: Containerization solution feature comparison [31, 32]

	Automatic Scaling	**Load balancing**	**High availability**
Standalone Docker	No	No	No
Docker Compose	No	No	No
Docker Swarm	No	Yes	Yes
Kubernetes	Yes	Yes	Yes

Deployment strategy selection

As a major part of this thesis was deployment in Hradec Králové, a suitable deployment strategy had to be selected.

Manual Manual deployment consists of a developer or a system administration logging into the virtual machine over SSH, updating the deployment parameters and/or application versions, and manually invoking the `docker compose up` command. Due to the use of Docker Compose, the room for error with this approach is limited, as there are no modifications performed to the underlying system. On the other hand, the configuration is not versioned, so there is no way to keep history of the modifications, for example to allow rollbacks in case of failures.

Automated CI/CD As opposed to manual invocation of commands on the production machine, the process can be automated with the use of CI/CD to further minimize the room for error, and make the deployment process more streamlined, and also auditable. A CI/CD pipeline can be configured to automatically deploy the code when a commit is pushed to a predefined branch on GitLab. With such automated approach, the configuration of the deployment is stored within a git repository, and all secrets are stored in the form of variables in GitLab. [33]

In GitLab, the automated approach can be used to deploy to multiple parallel deployments of the same project, used for different purposes, with the help of the Environments feature. [33] The feature allows different variables to be used depending on the Environment, allowing clear separation between various deployments. [33]

Conclusion The automated approach was selected due to its useful features and advantages. Firstly, the deployment was intended to happen iteratively, allowing Hradec Králové to verify the changes in a non-production deployment before confirming their validity. Deploying such changes frequently would be potentially error-prone and tedious. Secondly, when deploying to production, the use of GitLab environments allows to use the same procedure as for development, while only using different variables. This further reduces the room for error.

Acceptance criteria

HK-D1: LKOD is deployed on the infrastructure of Hradec Králové in a virtual machine.

HK-D2: All relevant parts of LKOD, namely Catalog, Administration, API, SPARQL and FTP are exposed to the public internet.

HK-D3: Internal parts of LKOD like the PostgreSQL database and Redis are NOT exposed to the public internet.

HK-D4: The LKOD domain and its subdomains are using correctly configured HTTPS handled by the existing WAF at a higher level.

HK-D5: The optional FTP feature for storage of distribution files is enabled and usable.

HK-D6: The deployed LKOD instance has all improvements implemented in 4.1 LKOD administration user interface included.

HK-D7: LKOD is configured with a valid SMTP server of Hradec Králové in order to be able to send password reset emails.

HK-D8: The deployment contains a single user with the role `superadmin`, correctly configured with the email address of a responsible person from the Hradec Králové IT team.

HK-D9: All functionality of the deployed LKOD instance is tested using a manual testing approach.

For the manual testing defined in HK-D9, a number of test cases verifying the successful configuration and deployment of LKOD was designed. Note that the below test cases only test the deployment of LKOD, not its functionality. The actual functionality of the LKOD feature set is tested during development, and is therefore already covered, making it outside of the scope of the deployment task.

Test cases: LKOD Catalog

TC-CAT-1: The catalog is accessible on the designated public URL.

TC-CAT-2: Given an organization and a dataset was added via LKOD Admin, then they are displayed in the catalog user interface.

TC-CAT-3: Given a file was uploaded in LKOD Admin, when the relevant distribution download link is pressed, then the file is downloaded.

TC-CAT-4: Given a file was uploaded to an external location and registered in LKOD Admin, when the relevant distribution download link is pressed, then the file is downloaded.

TC-CAT-5: When the "Přihlásit se" button in the bottom of the page is pressed, then the user is navigated to the LKOD Admin website.

Test cases: LKOD Administration

TC-ADM-1: The administration is accessible on the designated public URL.

TC-ADM-2: Given a user is logged in and an organization they belong to exists, when they fill in the dataset creation form and hit save, then they are navigated back to the list of datasets and the new dataset is present.

TC-ADM-3: Given a user is logged in and a dataset they are the owner of exists, when they upload a distribution file to the dataset, then the file appears in the list of distributions for that dataset with a working link.

4.3.3 Training

Acceptance criteria

HK-T1: A responsible person from Hradec Králové IT team was lead through the process of adding a user, organization, dataset, and a distribution in the LKOD administrator user interface.

HK-T2: A responsible person from Hradec Králové IT team was lead through the process of registering the LKOD instance in NKOD.

5 Implementation

5.1 Organization & user management in LKOD

5.1.1 Superadmin role

As discovered during analysis, the addition of user and organization management into LKOD administration necessitates a new user role, `superadmin`. The addition of the `superadmin` role requires modifications in three places: the database, the LKOD backend, and the LKOD administration frontend. Each of those changes are discussed below.

Database

As for the database, the following database UP migration was added, to extend the pre-existing `enum_user_role` enumeration.

```
ALTER TYPE "enum_user_role" ADD VALUE 'superadmin' AFTER '
    admin';
```

Every UP migration needs to be matched with a DOWN migration, which has turned out to be non-trivial due to the fact that PostgreSQL does not support removal of enumeration constants[1], only addition. The removal needed to be performed in a round-about way, with the use of a temporary enumeration, and migration via a string interpretation of the enumeration value. The DOWN migration SQL is as follows:

```
UPDATE "user"
SET "role" = 'admin'
WHERE "role" = 'superadmin';

-- The only way to remove an enum value is via a temporary
    enum
CREATE TYPE "enum_user_role_tmp"
    AS ENUM ('user', 'admin');

ALTER TABLE "user"
    ALTER COLUMN "role" DROP DEFAULT;
```

[1] https://www.postgresql.org/message-id/21012.1459434338%40sss.pgh.pa.us

```
ALTER TABLE "user"
    ALTER COLUMN "role" TYPE "enum_user_role_tmp"
        USING ("role"::text::"enum_user_role_tmp");

DROP TYPE "enum_user_role";

ALTER TYPE "enum_user_role_tmp"
    RENAME TO "enum_user_role";
ALTER TABLE "user"
    ALTER COLUMN "role" SET DEFAULT 'user'::"enum_user_role"
        ;
```

Backend

On the backend side, existing code needed to be adjusted in nearly all Controller classes to allow a `superadmin` to manage all resources in the system. Additionally, the response of the `POST /login` endpoint was adjusted to include the `role` field so that the user interface can later display additional user interface elements if the current user is `superadmin`.

The OpenAPI, the `docs/user_roles.md` documentation, and test cases were updated accordingly. In the OpenAPI documentation, the `LoginOutput` data was extended in the following way:

```
role:
    type: string
    enum: [superadmin, admin, user]
```

Administration frontend

On the frontend side, the `NextAuthOptions` object was updated to populate the `role` field, which was added to the `Session`, `User`, and `LoginOutput` interfaces, later to be used by code which adds user and organization management. No changes visible to the user were performed.

5.1.2 User management

The first feature with a visible impact in the user interface is the addition of user management. This feature necessitated modifications

in both the backend and the administration frontend; no changes were needed in the database. The user interface has been designed to precisely match the style of the existing user interface, and to be fully responsive, including mobile screen sizes. The implemented user management user interface is depicted in Appendix B.

Backend

To support user management, five new endpoints have been implemented in the *LKOD Backend* project. Authentication enforces the `superadmin` role for each of the endpoints, otherwise a HTTP code 403 Forbidden is returned. The newly added endpoints are listed and described in Table 5.1.

Table 5.1: User Management REST API Endpoints

Method	Path	Endpoint Description
`GET`	`/users`	Retrieves the list of users
`POST`	`/users`	Adds a new user
`GET`	`/users/me`	Returns the current user's information
`PATCH`	`/users/{userId}`	Update a user
`DELETE`	`/users/{userId}`	Deletes a user

5.1.3 Organization management

Similarly to the user management endpoints, to support organization management, five new endpoints have been implemented in the *LKOD Backend* project. Authentication enforces the `superadmin` role for each of the endpoints, otherwise a HTTP code 403 Forbidden is returned. The newly added endpoints are described in Table 5.2.

In addition to the implementation of the new endpoints, the existing endpoint `PATCH /organizations/{organizationId}` has been modified so that the field `arcGisFeed` is returned as long as the calling user possesses the role `superadmin`. The presence of the `arcGisFeed` signifies that the datasets of the organization are fetched automatically from ArcGIS instead of being manually created and managed

in LKOD. The `arcGisFeed` is needed in the frontend side so that the organization modification user interface field can be pre-filled with the current value, which was not necessary until now, explaining the current non-presence of the `arcGisFeed` field.

Table 5.2: Organization Management REST API Endpoints

Method	Path	Endpoint Description
`POST`	`/organizations`	Adds a new organization
`PATCH`	`/organizations/{organizationId}`	Updates an organization
`DELETE`	`/organizations/{organizationId}`	Deletes an organization
`POST`	`/organizations/{organizationId}/members/{userId}`	Adds a user into an organization
`DELETE`	`/organizations/{organizationId}/members/{userId}`	Removes a user from an organization

5.1.4 Dataset ownership transfer

After implementation of the user management feature, it was now possible to delete users. However, upon deletion of a user who was the owner of one or more datasets (signified by the foreign key `userId` in the `datasets` table), all of those datasets would be cascade-deleted[2] as well. The described shortcoming necessitated the implementation of an additional feature where the owner of each dataset could be changed, so that the datasets can be migrated to a different user before the deletion of their former owner.

Backend

In order to support the described feature in the backend part of the solution, the endpoint `PATCH /datasets/{datasetId}` was modified to accept a `userId` field in its request body. The `userId` field is only accepted as long as the current user possesses the `superadmin` role, otherwise the request is rejected with a 403 Forbidden response. This

[2]SQL cascade delete automatically deletes dependent records in child tables when a record in the parent table is deleted.

additional condition is needed because normally, the endpoint is accessible also to users carrying the `user` and `admin` role, where `user` can update datasets they own, and `admin` can update datasets that belong to the same organization as they do.

The OpenAPI documentation, and test cases were updated accordingly. In the OpenAPI documentation, the `UpdateDatasetInput` was extended in the following way:

```
UpdateDatasetInput:
# ...omitted for brevity
userId:
    type: number
# ...omitted for brevity
```

Administration frontend

On the frontend side, in the datasets view, each item has been extended with a "Vlastník" ("Owner") label, displaying the name of the current owner of a dataset. Additionally, if the current user is a `superadmin`, a "Vlastnictví" ("Ownership") button is displayed for each dataset. Pressing the button displays a modal dialog allowing the owner to be changed to any user in the system. The implemented user management user interface is depicted in Appendix D.

5.2 CLM maintenance

CLM is currently deployed at migrace.opendata.praha.eu, allowing any organization to migrate their datasets from CKAN to LKOD without the need to install CLM on their own infrastructure. The existing deployment of the application works well, as long as the version of CKAN that it is being used with generates API tokens that are less or equal 50 characters long.

In order to improve the user experience of CLM usage, modifications were performed and submitted into the source code repository in the form of merge requests[3,4], reviewed, and merged. The modified

[3] `https://gitlab.com/operator-ict/golemio/lkod/CLM/-/merge_requests/4`

[4] `https://gitlab.com/operator-ict/golemio/lkod/CLM/-/merge_requests/5`

version of CLM can also be found in Appendix G. Now follows discussion about every performed modification, solving shortcomings identified in the section 4.2.

5.2.1 CKAN access token is too long

At the time when CLM was built, CKAN has been using the same authentication mechanism for external API requests as it does now. However, since the time when CLM was built and deployed, the length of the CKAN API key has increased substantially. At the time of writing, the generated token was 175 characters in length, which does not fit into the designated field capped at 50 characters.

The issue was addressed by increasing the `CKAN_api_key` field length limit to 256 characters in the `app/src/models/LoginForm.py` file.

```
CKAN_api_key = StringField('API klic CKAN', validators=[
    Length(0, 256)])
```

5.2.2 Internal Server Error when loading datasets

When running the CLM application locally, a crash occurred when attempting to load a dataset from CKAN. The stack trace lead to inside the `jsonschema` library, which was being invoked in the `app/src/components/clm/JSONValidator.py` file.

The fact that the crash did not occur in production, which was using code from the same commit, narrowed down the cause to few possible variables like the environment, or the packages that were being used. The latter turned out to be the case. The version of the `jsonschema` Python library was not specified in `requirements.txt`, which means the latest version was pulled during each invocation of `pip install`.

The issue was remedied by identifying and locking the last working version of the library in the `requirements.txt` file. The last working version of the `jsonschema` library was identified to be be `4.17.3`. In the GitLab CI/CD job history of the CLM project [5], it can be verified

[5] https://gitlab.com/operator-ict/golemio/lkod/CLM/-/jobs/3032888085

that the last production image was built with a not yet affected version `4.16.0`, confirming the validity of the fix.

5.2.3 Data leak in the CLM user interface

In the CLM code, Python class variables are used incorrectly, leading to a number of variables being shared among class instances where not appropriate, and possibly not intended. This manifests in confusing behavior where revealing the dataset error dialog in the user interface leads to duplication of errors from the previous times the dialog was opened.

The issue was fixed by turning all class variables into instance variables in the files `app/src/models/JSONValidator.py` and `app/src/components/clm/JSONValidator.py`.

The problematic code was changed from

```
class JSONValidator:
    schema = None
    resolver = None
    api_url = ""
    errors = []

    def __init__(self, api_url):
        self.api_url = api_url
        self.schema = requests.get(ommited).json()
```

to

```
class JSONValidator:

    def __init__(self, api_url):
        self.api_url = api_url
        self.schema = requests.get(ommited).json()
        self.errors = []
```

5.2.4 Migrate HTTP distribution links to HTTPS

During automatic dataset distribution migration, CLM copies the distribution files from the CKAN storage to the LKOD FTP storage. However, if the distribution file is hosted externally outside of the CKAN instance, the link to the distribution file is simply copied instead.

It was observed that municipalities like Hradec Králové sometimes use HTTP links to file even though the file is also accessible over HTTPS.

Since HTTPS is required in the dataset creation form[6] according to the best practices, a dataset which uses a HTTP link is not considered valid, and displays a warning [24].

The improvement is implemented in the form of a modification of CLM, which intelligently replaces HTTP with HTTPS as long as the files are also accessible using HTTP.

The following Python function was introduced and applied to each external file URL during migration:

```
def use_https_if_ok(resource):
    if "http://" not in resource:
        return resource

    https_resource = resource.replace("http://", "https://")
    try:
        response = requests.get(https_resource)
    except requests.ConnectionError:
        return resource

    return https_resource if response.status_code == 200
       else resource
```

5.2.5 User interface improvement

A minor improvement was done to the user interface, making it easier for the user to orient themselves within the list of displayed errors during a dataset migration. When displaying the modal with a list of errors, instead of a long string spanning over multiple lines, each error is now displayed as a separate list item in a list of errors.

[6] https://data.gov.cz/formulář/registrace-datové-sady

5.3 LKOD deployment in Hradec Králové

5.3.1 Modification

Before deploying LKOD in Hradec Králové, customization of the page needed to be carried out. This customization was performed according to a new Hradec Králové visual style, to which the municipality is in the process of migrating. [34] A style guide, the custom "Hradec" font, a logo, and background images to use were provided as the basis for the customization.

During the implementation of the Hradec Králové visual style, changes were initially communicated with the Hradec Králové IT team using emails and a shared Word document. However, due to its disorganized nature, discussions were soon moved to GitLab in the form of issues, which allowed efficient tracking of additional modification requests and reporting of bugs.

The modified version of LKOD for Hradec Králové is available in a public GitLab group [7], and can also be found in Appendix G.

Administration

The *LKOD Administration* user interface was modified according to the Hradec Králové visual style guide, the footer has been updated with a logo of the Laboratory of Service Systems, and the social links were updated to point to the social networks of Hradec Králové. No other modifications were performed. The resulting user interface is depicted in Appendix E.

Catalog

Similarly to the aforementioned changes, *LKOD Catalog*'s user interface was also modified according to the Hradec Králové visual style guide, the laboratory logo was added, and the social links update. However, apart from these changes, the content on the pages was also modified. Specifically, all pages under "O otevřených datech" ("About open data") and "O katalogu" ("About the catalog") were updated with paragraphs provided by Hradec Králové. The first three examples on

[7] `https://gitlab.com/mmhk-opendata/lkod`

the "Využití a přínos otevřených dat" ("Use and benefit of open data") were replaced by applications specific to the Hradec Králové region. The "O katalogu" ("About the catalog") page has had a contact form inserted, and a link to the contact section was added to the footer of all pages. The "Data Praha" link has been replaced with "Hradec v datech", which is a website with an identical purpose, but specific to Hradec Králové. Lastly, a popup asking for feedback leading to a Prague-specific form has been removed. The resulting user interface is depicted in Appendix F.

5.3.2 Deployment

Deployments is performed using SSH with public key authentication, executed by a GitLab CI/CD pipeline. The necessary files are copied over SSH, after which the necessary Docker Compose commands are executed.

The persistent state of LKOD consisting of the PostgreSQL database, the SPARQL database, and the uploaded files, is stored in Docker Volumes[8]. To facilitate backups, a backup and a restore command were prepared in the *LKOD Deployment* repository[9].

Two deployment environments were created: a `development` environment used during implementation, and a `production` environment activated after the modifications have been approved. The majority of the pipeline configuration was shared, and the remaining difference between the environments has been captured in the form of two GitLab environments[10]. Deployment parameters of non-sensitive nature were captured using separate `docker-compose.yml` override files[11], where the correct override is selected during deployment depending on the environment.

A deployment guide has been created in the form of a README file in the *LKOD Deployment* repository, allowing easy re-creation of

[8]Docker Volumes are the preferred mechanism for persisting data generated by and used by Docker containers. [35]

[9]`https://gitlab.com/mmhk-opendata/lkod/lkod-deployment`

[10]GitLab environments offer tracking of the deployment state, and a clear separation of variables between different environments. [33]

[11]A Docker Compose override file allows for extension of an existing Compose file, with the possibility to override selected parameters. [36]

the deployment in the future. It can therefore lead as a base for other municipality interested in the deployment of LKOD, thanks to the isolated nature of the deployment parameters.

Development deployment

For continuous evaluation of implemented changes, a development deployment was set up on the virtual machines of Laboratory of Service Systems (SeSLab), which was continually updated with incorporated feedback from Hradec Králové. The deployment resides on the domain lkod.duckdns.org, and it set up with a testing account, with the email set to *superadmin@lkod.cz*, password *"pass"*. Although originally a DNS record in the Faculty of Informatics (FI) was supposed to be used, due to difficulties during communication with the FI system administrators, a free DNS service was used as a fallback. This fallback domain proved to be no issue, and it has shown to be sufficient for testing purposes.

For password reset testing purposes, a fake SMTP server is available at emails.lkod.duckdns.org, where is is possible to temporarily view any emails sent by LKOD, without the need for a full SMTP server.

The development deployment is automatically updated with the latest changes from the `development` branch in each of the LKOD repositories using a GitLab CI/CD pipeline, allowing rapid iteration without the need for manual re-deployment. A deployment strategy closely matching production was selected to minimize the potential differences issues specific to production.

Production deployment

After all changes were approved by the Hradec Králové IT team in the development environment, a production deployment followed. The deployment was performed in the Hradec Králové infrastructure, and the public domain opendata.mmhk.cz has been allocated and used.

The deployment to production was a matter of performing the documented initial steps, configuring the deployment GitLab CI/CD variables according to the documentation, and then launching a pipeline.

Automatic deployment is then performed from the `master` branches of each LKOD project.

User accounts carrying the *superadmin* role were created for the members of the Hradec Králové IT team, and the initial organization "Statutární město Hradec Králové" was inserted. With this organization in place, the first dataset "Úřední deska Hradec Králové" was created.

5.3.3 Testing

After successful deployment in the production environment, manual testing was performed using the manual testing strategy designed as a part of the section 4.3.2. All tests passed successfully, confirming that the deployment was stable and ready for use.

5.3.4 Training

A meeting has been set up with the members of the Hradec Králové IT team, where they were lead through every available feature in the system. They were introduced to the GitLab group containing all forked projects, and given the *Owner* role. They were specifically notified of the *README* file containing instructions for the deployment, and the backup/restore procedure of all LKOD data.

5.3.5 Registration in NKOD

Upon successful verification that the deployment matches expectations, the final step is to register the Hradec Králové LKOD in NKOD. The registration was achieved by following the official documented procedure, specifically by filling out a registration form[12], downloading the generated file, and submitting it to the relevant Data Box. The registration was processed within a day, as the initially created dataset appeared in NKOD the next day, confirming the success of both the deployment and registration.

[12] `https://data.gov.cz/formul%C3%A1%C5%99/registrace-lok%C3%A1ln%C3%ADho-katalogu`

5.3.6 Challenges

During the deployment of LKOD in Hradec Králové, a number of challenges arose.

Low quality of existing data in CKAN

It was ultimately decided by the Hradec Králové IT team that the existing data in CKAN will not be migrated to LKOD using CLM, due to the reportedly disorganized nature of the existing catalog, and non-compliance with OFN. The result of the analysis by the Hradec Králové IT team was that it would take more time and effort to migrate the data than to properly enter the data again. Therefore, CLM was not used for migration from CKAN, and the migration of the data becomes out of scope of this thesis.

Domain usage issues

The initial idea of the Hradec Králové IT team was to place the LKOD subdomains under the domain `hradeckralove.org`, leading to an easily recognizable URL, such as `https://opendata.hradeckralove.org/`. However, during analysis, it was discovered that accessing the domain `hradeckralove.org` from the Hradec Králové infrastructure is problematic due to undisclosed technical reasons, pointing towards `hradeckralove.org` being outside of the Hradec Králové infrastructure.

As a result, it was decided that the LKOD subdomains should be placed under `mmhk.cz`, which is a domain belonging to the Municipality of Hradec Králové. This lead to another issue, where the subdomain of choice, `opendata.mmhk.cz`, was already taken up by the existing CKAN deployment. It was ultimately decided that the CKAN instance will be moved to `ckan.mmhk.cz`, and the `opendata.mmhk.cz` domain will be occupied by LKOD.

WAF interference

During production deployment, without an apparent reason, the components of *LKOD* failed to communicated with each other via their configured public URL, although such configuration worked well in

the development deployment. It turned out that internal requests from the *LKOD Catalog* to the SPARQL server were routed through the WAF (Web Application Firewall) of Hradec Králové. The WAF has observed that the requests from the SPARQL server contain URL parameters resembling SQL queries, and banned the host from communicating with itself under suspicion of an SQL injection attack. This problem was solved by configuring *LKOD Catalog* and *LKOD Administration Backend* to perform internal SPARQL requests via the internal Docker Compose network, rather than via the public URL. The remaining URLs linking the *LKOD* components together had to stay in their public form, because apart from being used for internal communication, they are also rendered in multiple parts of the user interface.

Invalid HTTPS configuration

During in-development testing of CLM with links from the CKAN deployment of Hradec Králové, it was discovered that the HTTPS configuration of Hradec Králové on the domain `mmhk.cz` was incorrect. The server has been sending a broken certificate chain, which included a wrong intermediary certificate and a wrong root certificate.

This situation is handled gracefully by modern web browsers using a technique called AIA chasing (or AIA fetching). In this technique, the web browsers look up and automatically download the rest of the certificate chain up to the trusted root certificate in order to verify the validity of the chain. [37]

However, some networking libraries, for example `requests` in Python, do not have this functionality built in, although it is currently being discussed [13]. As a result, when a request with such invalid certificate chain is performed using a network library without this support, it fails, often with an unclear error message. This relates to our solution in two ways.

Firstly, the automatic HTTP to HTTPS conversion mechanism implemented in the section 4.2.4 labels such links as not working, even though they technically could be used with modern web browsers.

Secondly, NKOD performs automatic checking of all provided distribution links once per day, displaying a red warning next to all

[13] `https://github.com/python/cpython/issues/62817`

links that do not work. Since NKOD internally uses a library without AIA chasing, it fails the same way as the `requests` Python library, and labels the link as not working, even though modern web browsers do open it without a warning. This would later manifest in the same behavior with actual LKOD distribution links, but it was fortunately caught early.

Upon comparison of the incorrect and the correct certificate chain, it was discovered that the non-matching certificates differed by name only slightly, "Thawte RSA CA 2018" vs. "Thawte TLS RSA CA G1" and "DigiCert Global Root CA" vs. "DigiCert Global Root G2", which might point towards this misconfiguration being a human error. Therefore, this issue was reported to the Hradec Králové IT team, and promptly fixed by replacing the broken certificate chain with a suggested correct one. The fix was verified to be correct by manual inspection of the `mmhk.cz` certificate chain, and also of the affected dataset[14] in NKOD.

[14] `https://data.gov.cz/dataset?iri=https%3A%2F%2Fdata.gov.cz%2Fzdroj%2Fdatov%C3%A9-sady%2F00268810%2F998562490`

6 Future work

After the successful deployment of LKOD in Hradec Králové, a number of improvements for the future of LKOD were identified during discussions with the Hradec Králové IT team. The list of improvements can be used as a basis for potential future work in the field of open data.

6.1 A "Recent additions" section in the LKOD catalog

To facilitate citizens' easy access to recently published data about their municipality, it would be beneficial for the LKOD catalog to offer a "Recent additions" ("Novinky") section on the home page. This section would contain a small number of datasets, ordered by the date of creation or update.

The data necessary for this functionality is already tracked in the LKOD Backend SQL database in the `createdAt` and `updatedAt` columns of the `datasets` table. However, the SPARQL endpoint that the LKOD Catalog uses to populate its user interface does not expose these columns since they are not a part of the actual dataset metadata.

Operátor ICT has confirmed that the optimal solution to this problem would be to enrich the dataset metadata with the necessary field(s), like the creation and modification date, which are already tracked in the database. As such fields are already a part of DCAT-AP 2.0.1 as optional fields of a dataset, specifically `dct:issued` and `dct:modified`, they could be implemented in a standard manner. [38]

6.2 Deeper integration with hradecvdatech.cz

Hradec Králové is in the process of developing a data portal similar to data.praha.eu, available in a work-in-progress state at the time of writing at hradecvdatech.cz. During the implementation of this thesis, multiple links to the portal were added to LKOD.

However, in order for the users of the portals to be able to find relevant data easier, it was suggested that deeper integration could be introduced between these two sites. The primary idea was to in-

clude datasets and organizations from LKOD in the results of the *hradecvdatech.cz* search field.

Since *hradecvdatech.cz* is currently implemented in Framer[1], the ways in which the site can be made interactive are limited. Framer does offer support for custom components that are written in React, however, it is unclear whether this feature is flexible enough to support this use case. [39]

6.3 English language in the catalog

In both DCAT-AP-CZ and the dataset creation form itself, in addition to Czech, all textual fields can be filled in in English. It would, therefore, make sense for the catalog user interface to have the possibility to be switched to English. Hradec Králové has shown interest in this feature, primarily to increase the accessibility of the platform, for example for citizens who are interested in data, but have not yet been able to learn the Czech language, or completely foreign users.

6.4 Safer deletion of users and organizations

Currently, deleting a user or an organization deletes all datasets that belong to them. This is potentially dangerous, and can lead to data loss. Due to the danger of this operation, a confirmation dialog asking to type in the name of the user/organization was implemented. However, it would still be safer to implement deletion of users and organizations in a soft-delete[2] manner.

A similarly safe solution would be to allow deletion of users and organizations, but only after all datasets have been transferred from them or deleted. As one of the features implemented was the possibility to transfer the ownership of a dataset to a different user, it is now possible to manually move all datasets away from a user before deleting them. However, checking if they have any remaining datasets

[1]Framer: A design tool used for creating both interactive prototypes, and simple production-ready interfaces. `https://www.framer.com/`

[2]Soft-delete is a technique where data is marked as deleted by updating a flag or status, rather than being permanently removed from the database, allowing for potential recovery or auditing.

assigned still requires to check each dataset in the system visually. The solution would be for the delete confirmation dialog to display a list of datasets that still belong to the user or organization, allowing deletion of the user or organization only after they have been transferred or deleted.

7 Conclusion

The outlined goals of this thesis were successfully achieved, providing significant insights into the field of open data and, most importantly, producing tangible results.

Firstly, user and organization management was successfully implemented in the Local Open Data Catalog (LKOD), reviewed, and accepted by its maintainer Operátor ICT. This crucial feature is included in the upstream repository of LKOD, greatly improving user experience of LKOD in any future deployments by removing the need to ever perform SQL queries in LKOD after the initial deployment. This change alone has a major impact by significantly lowering the barrier of entry for new organizations into an existing LKOD instance, encouraging publication of open data.

Secondly, maintenance in the CLM codebase was performed, solving all issues that would prevent future deployment and usage of the application. Although CLM was ultimately not used in Hradec Králové, the maintenance performed is still relevant, and is planned to be reviewed and accepted by Operátor ICT.

Lastly, the LKOD user interface has been heavily customized according to the requirements of Hradec Králové. A thorough deployment and testing plan was designed and successfully executed, deploying LKOD into production at opendata.mmhk.cz. After deployment, members of the Hradec Králové IT team have been trained, and the deployed LKOD has been registered in the National Open Data Catalog NKOD, concluding the success of the aforementioned efforts.

During deployment, a number of both common and unusual issues were encountered and subsequently solved. The solutions are discussed in the final parts of this thesis, which can prove to be helpful during future production deployments of LKOD.

In collaboration with the Hradec Králové IT team, multiple future improvements of LKOD were identified, which would further increase the usability and accessibility of LKOD. The improvements can be implemented in future work.

Additionally, an intangible yet invaluable outcome was the strengthened collaboration between Operátor ICT and the Laboratory of Service Systems (SeSLab), which has the potential to further contribute

to the evolution of open data in the Czech Republic. Last but not least, another valuable result is the establishment of a communication channel between the municipality of Hradec Králové and SeSLab, paving the ground for future opportunities, not only in the field of open data.

Bibliography

1. ČESKO. *Zákon č. 106/1999 Sb., o svobodném přístupu k informacím* [online]. Zákony pro lidi.cz, 1999. [visited on 2024-05-02]. Available from: https://www.zakonyprolidi.cz/cs/1999-106.
2. *Národní katalog otevřených dat na Portálu veřejné správy* [*The national open data catalog on the Public Administration Portal*] [online]. Ministerstvo vnitra České republiky, 2015 [visited on 2024-05-02]. Available from: https://www.mvcr.cz/clanek/zpravodajstvi-narodni-katalog-otevrenych-dat-na-portalu-verejne-spravy.aspx.
3. KLÍMEK, Jakub; NEČASKÝ, Martin. *Otevřená data blíže* [*A closer look at open data*] [online]. Digital a Information Agency, 2018-12-19 [visited on 2024-05-02]. Available from: https://opendata.gov.cz/informace:otev%C5%99en%C3%A1-data-bl%C3%AD%C5%BEe.
4. KLÍMEK, Jakub; MÍŠEK, Jakub. *Co jsou otevřená data?* [*What is open data?*] [online]. Digital a Information Agency, 2018-12-19 [visited on 2024-05-02]. Available from: https://opendata.gov.cz/informace:start.
5. KLÍMEK, Jakub. *Kontext otevřených dat* [*The context of open data*] [online]. Digital a Information Agency, 2018-12-19 [visited on 2024-05-02]. Available from: https://opendata.gov.cz/informace:kontext:start.
6. *Applications - Národní katalog otevřených dat* (*NKOD*) [online]. Digital and Information Agency [visited on 2024-05-02]. Available from: https://data.gov.cz/applications.
7. KLÍMEK, Jakub; NEČASKÝ, Martin. *Otevřená data blíže* [*A closer look at open data*] [online]. Digital a Information Agency, 2018-12-19 [visited on 2024-05-02]. Available from: https://opendata.gov.cz/informace:otev%C5%99en%C3%A1-data-bl%C3%AD%C5%BEe.
8. PECHA, Tomáš; NEČASKÝ, Martin; KLÍMEK, Jakub. *Rozhodnutí o způsobu katalogizace otevřených dat* [*Deciding on the way to catalog open data*] [online]. Digital a Information Agency, 2016-07-08 [visited on 2024-05-02]. Available from: https://opendata.gov.cz/cinnost:rozhodnuti-o-zpusobu-katalogizace-dat.

9. KLÍMEK, Jakub. *Příprava katalogizačního záznamu o datové sadě* [*Preparation of a cataloging record of the data set*] [online]. Digital a Information Agency, 2018-03-23 [visited on 2024-05-02]. Available from: https://opendata.gov.cz/cinnost:priprava-katalogizacniho-zaznamu.

10. *DCAT Application profile for data portals in Europe* (*DCAT-AP*) [online]. European Union [visited on 2024-05-16]. Available from: https://op.europa.eu/en/web/eu-vocabularies/dcat-ap.

11. *Otevřené formální normy* (*OFN*) [*Open Formal Norms* (*OFN*)] [online]. Ministerstvo vnitra České republiky [visited on 2024-05-16]. Available from: https://data.gov.cz/ofn/.

12. *Rozhraní katalogů otevřených dat: DCAT-AP-CZ* [*Open data catalog interface: DCAT-AP-CZ*] [online]. Ministerstvo vnitra České republiky [visited on 2024-05-05]. Available from: https://ofn.gov.cz/rozhran%C3%AD-katalog%C5%AF-otev%C5%99en%C3%BDch-dat/2021-01-11/.

13. ŠKODA, Petr. *Rozhraní GraphQL Národního katalogu otevřených dat* [*National Open Data Catalog GraphQL interface*] [online]. Digital a Information Agency, 2022-03-14 [visited on 2024-05-02]. Available from: https://data.gov.cz/%C4%8Dl%C3%A1nky/rozhran%C3%AD-graphql-n%C3%A1rodn%C3%ADho-katalogu-otev%C5%99en%C3%BDch-dat.

14. NEČASKÝ, Martin. *Série Znalostní grafy: Díl 3: SPARQL* [*Knowledge Graphs Series: Volume 3: SPARQL*] [online]. Digital a Information Agency, 2021-01-25 [visited on 2024-05-02]. Available from: https://data.gov.cz/%C4%8Dl%C3%A1nky/znalostn%C3%AD-grafy-03-sparql.

15. KLÍMEK, Jakub. *Důležité pojmy v oblasti otevřených dat* [*Important terms in the field of open data*] [online]. Digital a Information Agency, 2016-10-12 [visited on 2024-05-02]. Available from: https://opendata.gov.cz/informace:d%C5%AFle%C5%BEit%C3%A9-pojmy-v-oblasti-otev%C5%99en%C3%BDch-dat.

16. BAPTISTA, Vitor et al. *What is CKAN?* [online]. The CKAN Association [visited on 2024-05-03]. Available from: https://docs.ckan.org/en/2.10/user-guide.html#what-is-ckan.

17. BAPTISTA, Vitor et al. *API guide* [online]. The CKAN Association [visited on 2024-05-03]. Available from: https://docs.ckan.org/en/2.10/api/.

18. BAPTISTA, Vitor et al. *Adding a new dataset* [online]. The CKAN Association [visited on 2024-05-03]. Available from: https://docs.ckan.org/en/2.10/user-guide.html#adding-a-new-dataset.

19. BAPTISTA, Vitor et al. *Customizing dataset and resource metadata fields using IDatasetForm* [online]. The CKAN Association [visited on 2024-05-03]. Available from: https://docs.ckan.org/en/2.10/extensions/adding-custom-fields.html.

20. *Rešerše vývoje standardů v oblasti katalogizace otevřených dat* [*Research on the development of standards in the field of open data cataloging*] [online]. Ministerstvo vnitra České republiky, 2018-12-19 [visited on 2024-05-03]. Available from: https://opendata.gov.cz/_media/dokumenty:c3v1_re%C5%A1er%C5%A1e_v%C3%BDvoje_standard%C5%AF_v_oblasti_katalogizace_otev%C5%99en%C3%BDch_dat.pdf.

21. KLÍMEK, Jakub. *Referenční implementace Lokálního katalogu otevřených dat (LKOD) [Local Catalog of Open Data (LKOD) reference implementation]* [online]. The CKAN Association [visited on 2019-01-18]. Available from: https://github.com/datagov-cz/lkod.

22. SKÁLA, Dominik. *Dokončení procesu vývoje serveru otevřených dat*. Brno, 2022. Available also from: https://is.muni.cz/th/h8118/. Diplomová práce. Masarykova univerzita, Fakulta informatiky. Supervised by Leonard WALLETZKÝ.

23. GONZALEZ, Gerardo; DAY, Janette, et al. *DKAN Documentation: DKAN Overview* [online]. CivicActions [visited on 2024-05-15]. Available from: https://dkan.readthedocs.io/en/latest/introduction/overview.html.

24. KLÍMEK, Jakub. *Přístup přes Nezabezpečený Protokol HTTP či FTP* [*Access via Unsecured HTTP or FTP protocol*] [online]. Otevřená data, 2019-07-02 [visited on 2024-04-26]. Available from: https://opendata.gov.cz/%C5%A1patn%C3%A1-praxe:http.

25. *What is Containerization?* [online]. Amazon Web Services [visited on 2024-05-16]. Available from: https://aws.amazon.com/what-is/containerization/.

26. POWELL, Ron. *Benefits of containerization* [online]. CircleCI, 2023-04-07 [visited on 2024-04-29]. Available from: https://circleci.com/blog/benefits-of-containerization/.

27. *How Compose works* [online]. Docker Inc. [visited on 2024-04-29]. Available from: https://docs.docker.com/compose/compose-application-model/.

28. PRASAD, Aanand. *Ways to set environment variables with Compose* [online]. Docker Inc., 2016-05-23 [visited on 2024-04-29]. Available from: https://docs.docker.com/compose/environment-variables/set-environment-variables/.

29. WALLEN, Jack. *Simplifying the mystery: When to use docker, docker-compose, docker swarm and Kubernetes* [online]. TechRepublic, 2021-03-16 [visited on 2024-04-27]. Available from: https://www.techrepublic.com/article/simplifying-the-mystery-when-to-use-docker-docker-compose-and-kubernetes/.

30. PRASAD, Aanand. *Use Compose in production | Docker Docs* [online]. Docker Inc., 2015-04-04 [visited on 2024-04-27]. Available from: https://docs.docker.com/compose/production/.

31. ISAIAH, Ayooluwa. *Setting up Docker Swarm High Availability in Production* [online]. Better Stack, 2023-12-22 [visited on 2024-04-27]. Available from: https://betterstack.com/community/guides/scaling-docker/ha-docker-swarm/.

32. HANNAFORD, Jamie. *Creating Highly Available Clusters with kubeadm* [online]. The Kubernetes Authors, 2018-01-12 [visited on 2024-04-27]. Available from: https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/high-availability/.

33. PUNDSACK, Mark et al. *Environments and deployments* [online]. GitLab.org [visited on 2024-05-06]. Available from: https://docs.gitlab.com/ee/ci/environments/index.html.

34. *Hradec Králové má novou vizuální identitu* [*Hradec Králové has a new visual identity*] [online]. Vinklář, Petr, 2024-03-06 [visited on 2024-05-18]. Available from: https://www.hradeckralove.org/hradec-kralove-ma-novou-vizualni-identitu/d-83915.

35. TURNBULL, James; STANLEY-JONES, Misty, et al. *Volumes | Docker Docs* [online]. Docker Inc. [visited on 2024-05-15]. Available from: https://docs.docker.com/storage/volumes/.

36. *Ways to set environment variables with Compose* [online]. Docker Inc. [visited on 2024-05-15]. Available from: https://docs.docker.com/compose/multiple-compose-files/merge/.

37. LYNCH, Vincent. *AIA Fetching: Solving a Common SSL Misconfiguration* [online]. Hashed Out by The SSL Store, 2016-10-17 [visited on 2024-04-27]. Available from: https://www.thesslstore.com/blog/aia-fetching/.

38. *DCAT Application profile for data portals in Europe (DCAT-AP): DCAT-AP 2.0.1* [online]. European Commission, 2021 [visited on 2024-05-19]. Available from: https://joinup.ec.europa.eu/collection/semic-support-centre/solution/dcat-application-profile-data-portals-europe/release/201-0.

39. *Framer for Developers: Code Components* [online]. Framer [visited on 2024-05-10]. Available from: https://www.framer.com/developers/#code-components.

A Dataset creation form by data.gov.cz

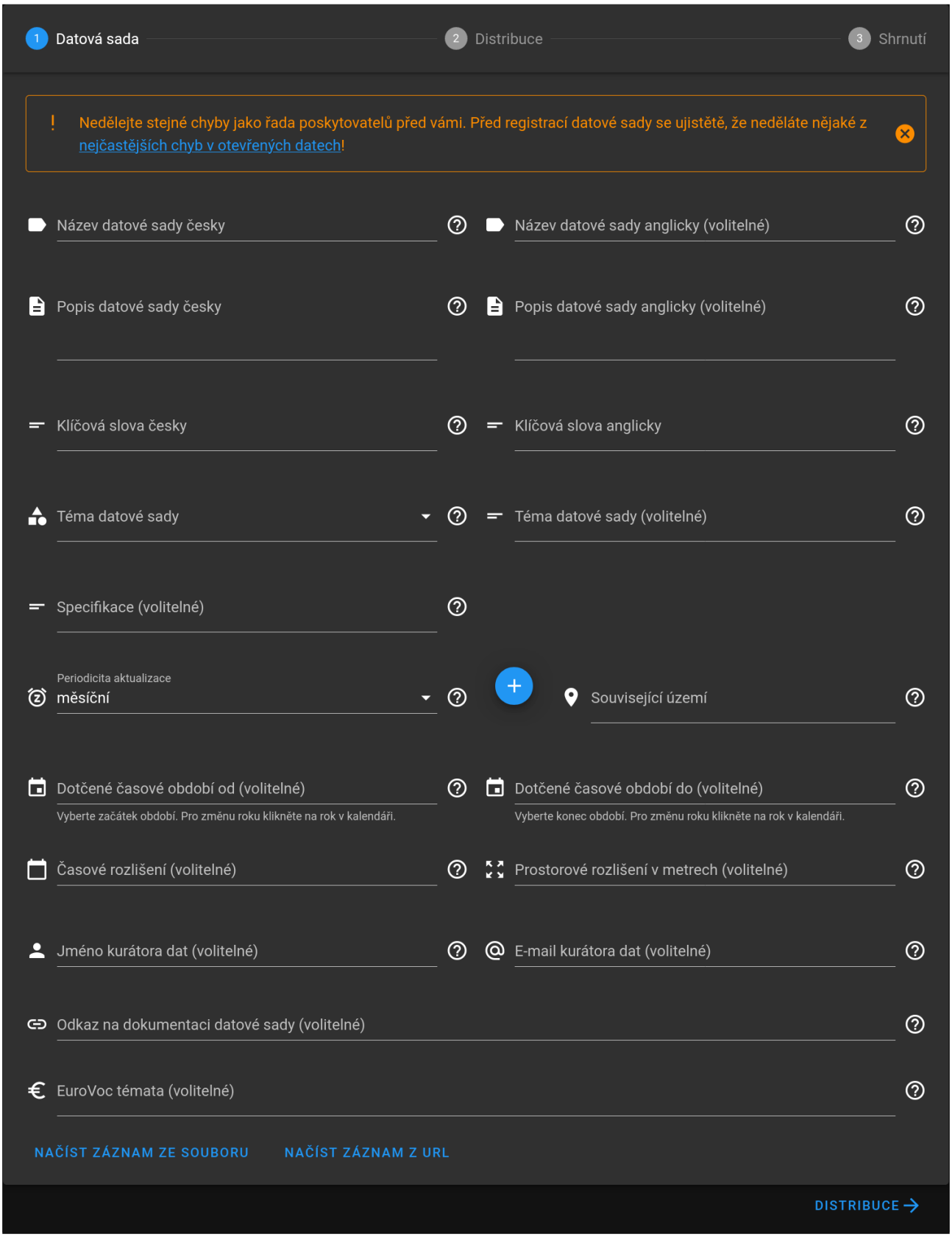

Figure A.1: Dataset creation form: Dataset information

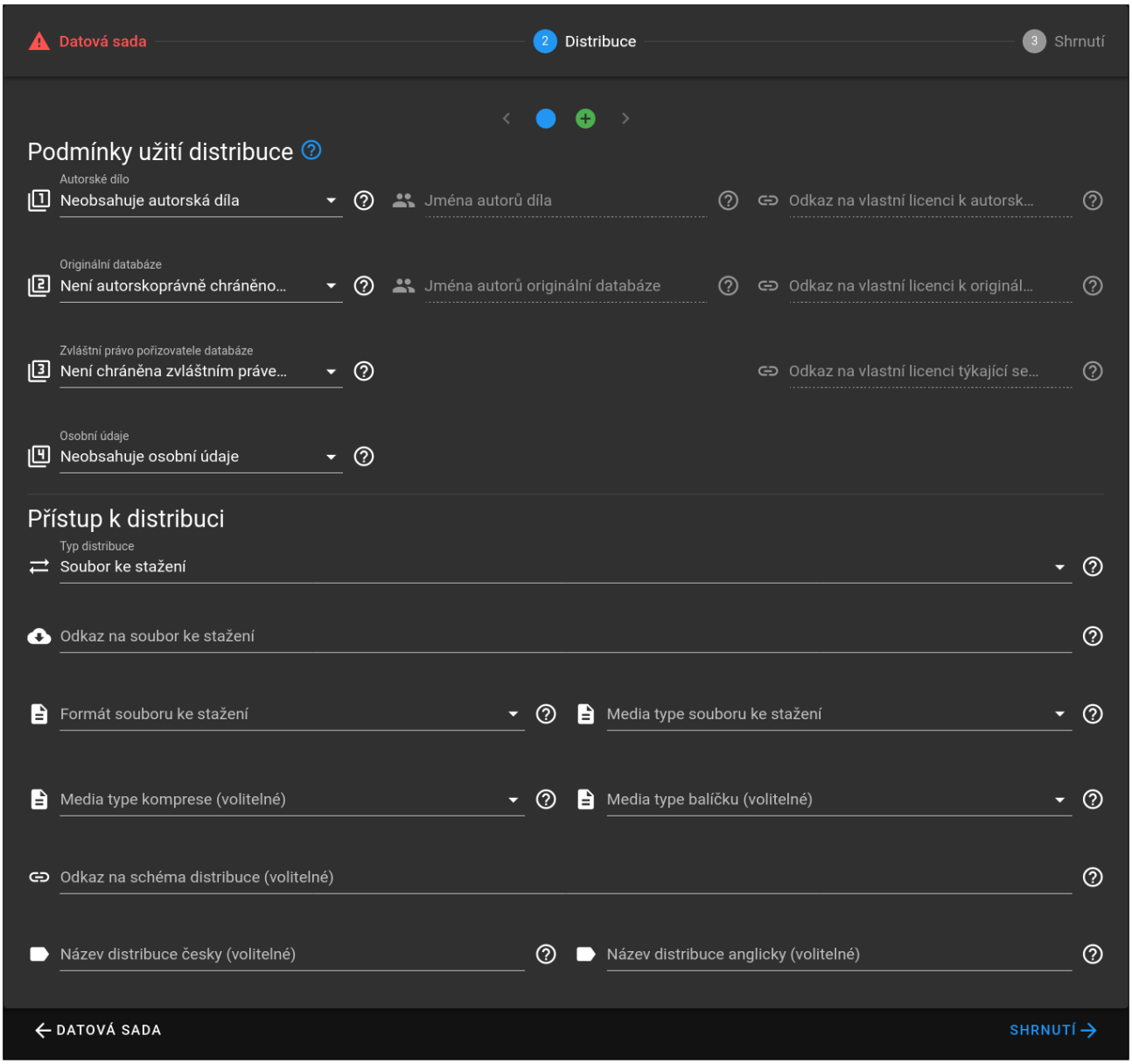

Figure A.2: Dataset creation form: Distribution information

B User Management

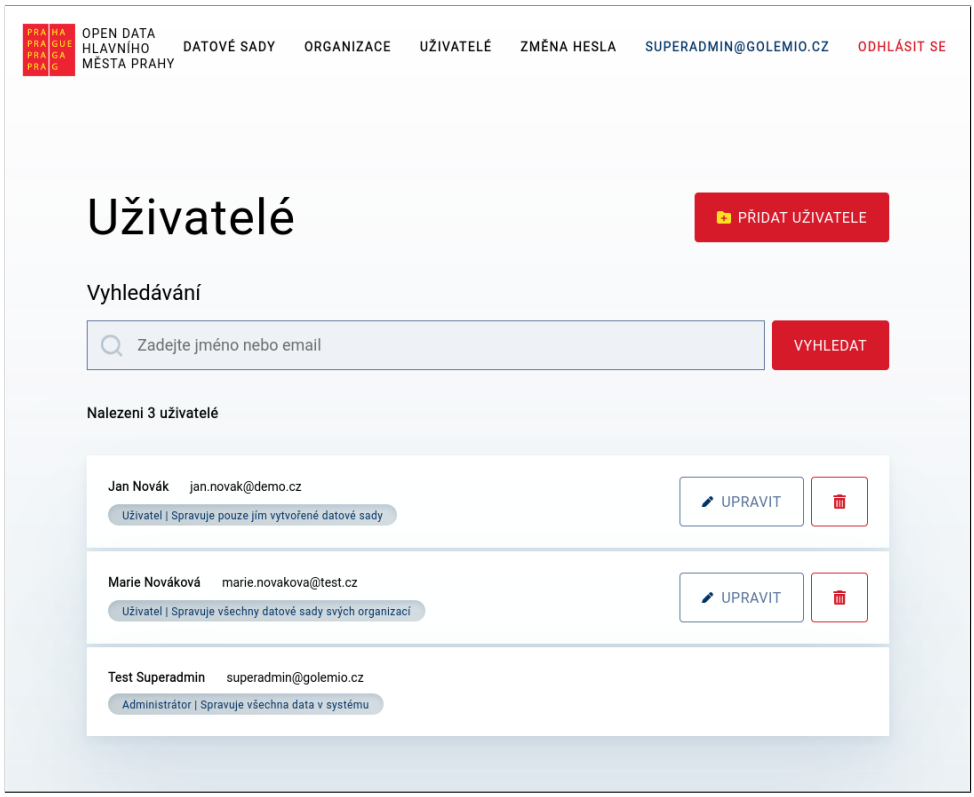

Figure B.1: List of users

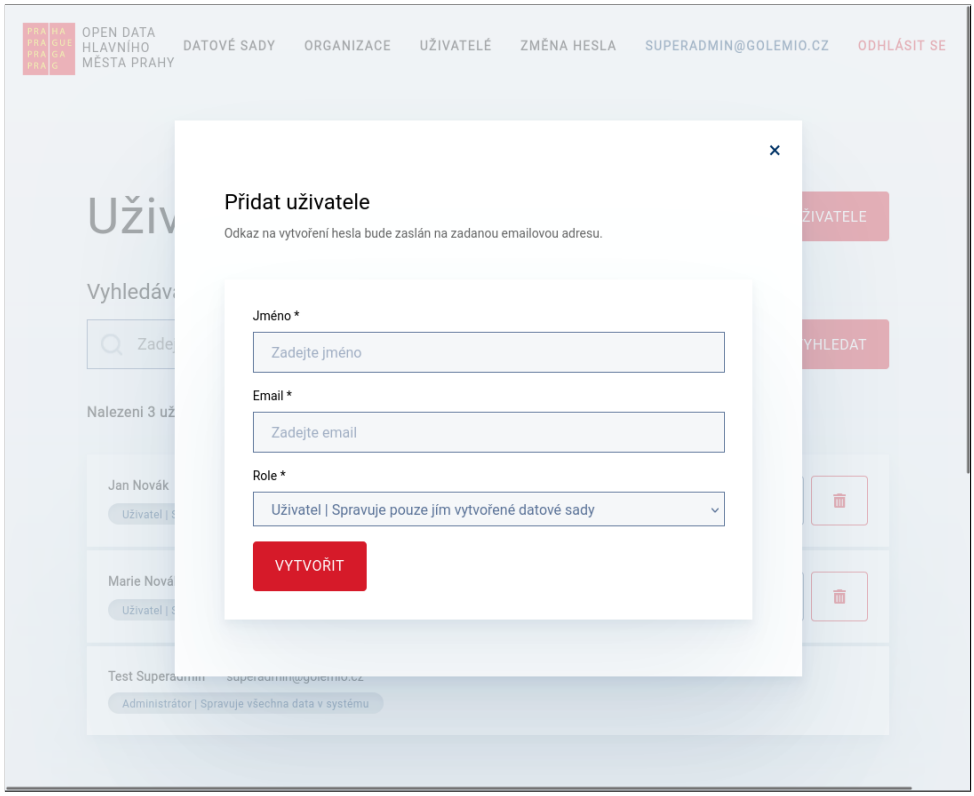

Figure B.2: Add user dialog

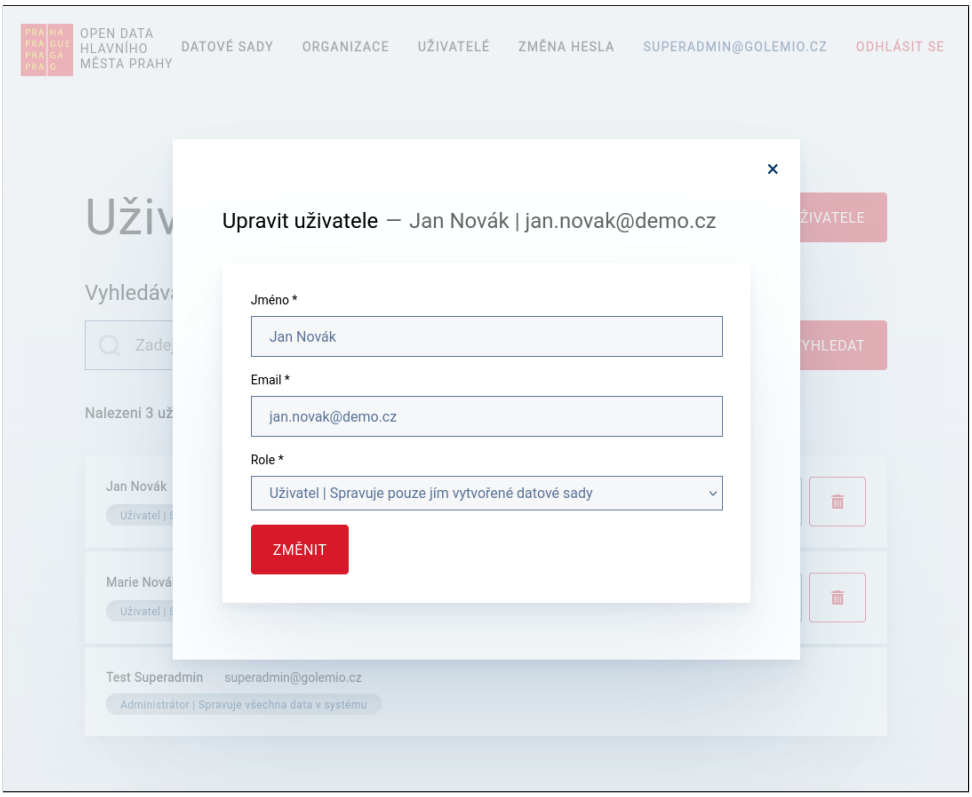

Figure B.3: Edit user dialog

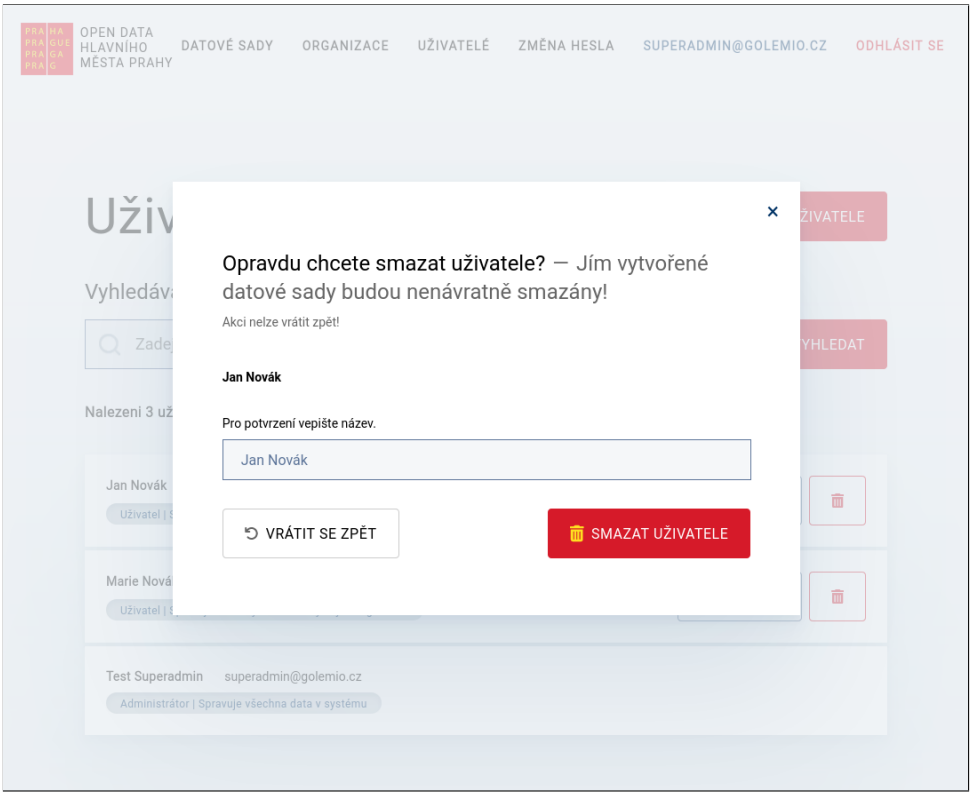

Figure B.4: Delete user confirmation

C Organization Management

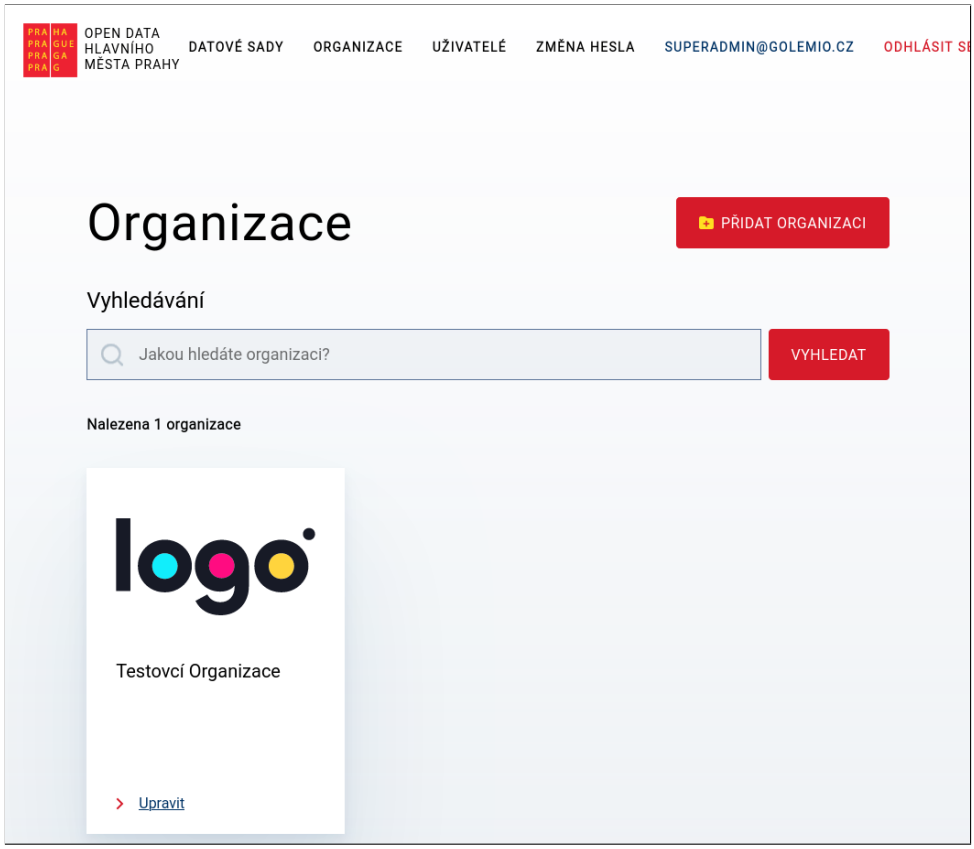

Figure C.1: List of organizations

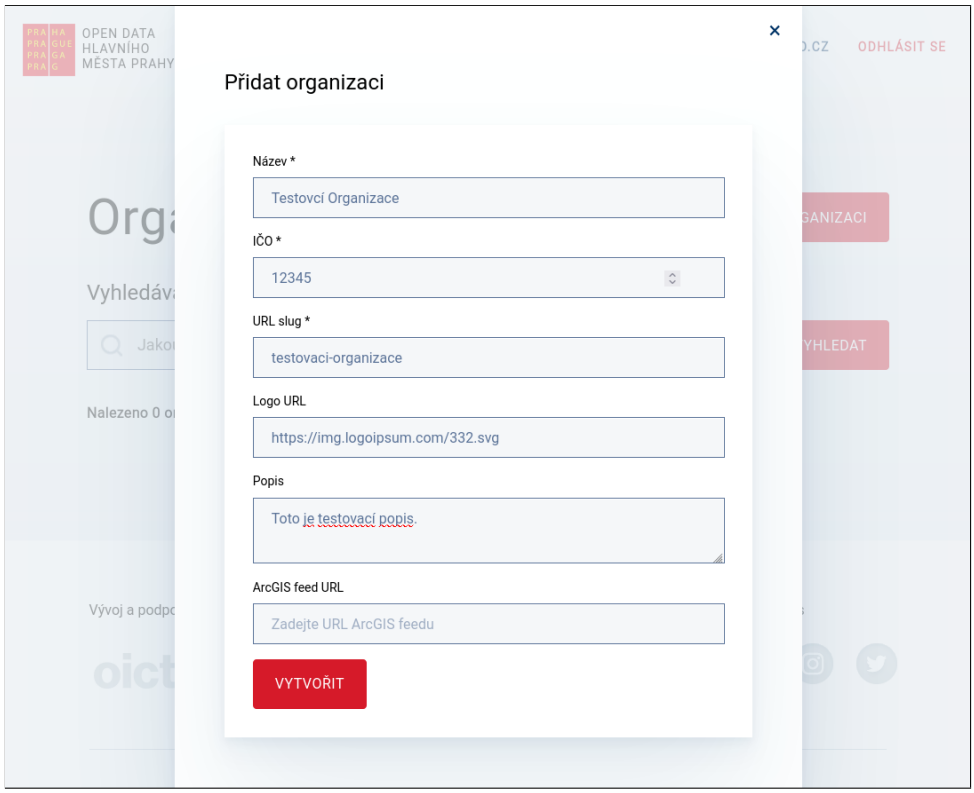

Figure C.2: Add organization dialog

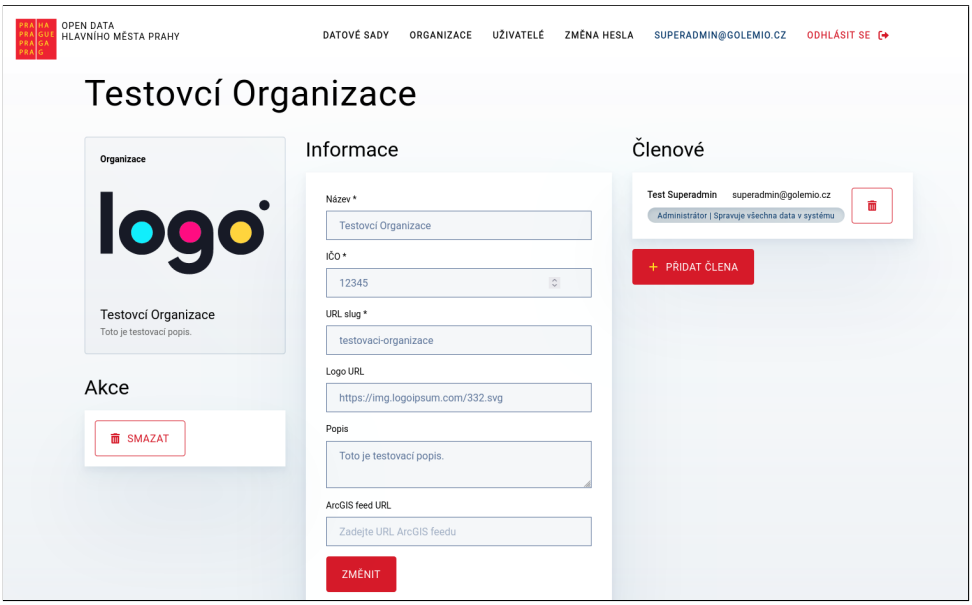

Figure C.3: Organization details

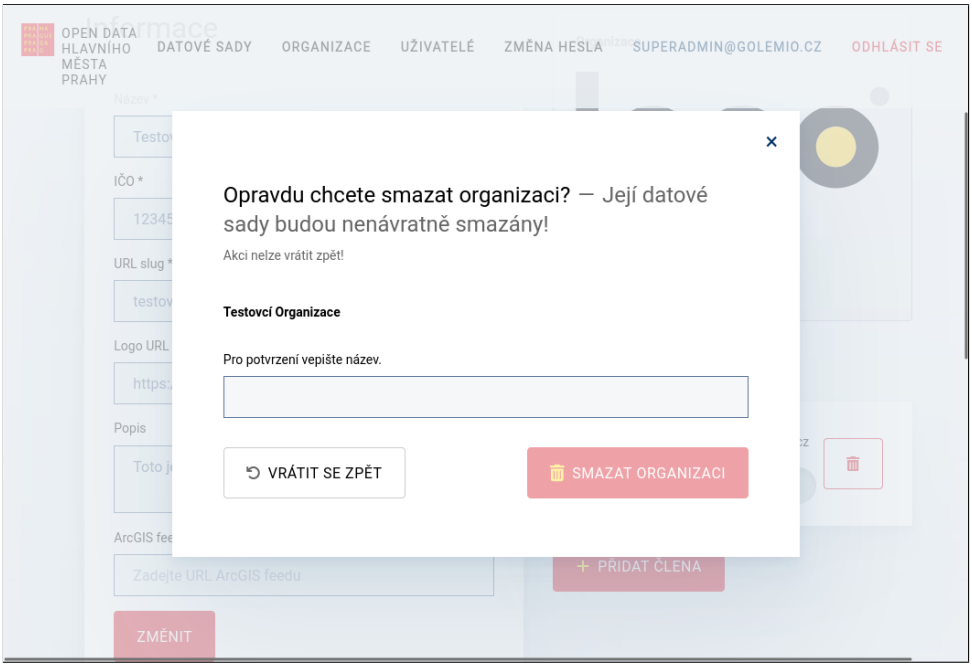

Figure C.4: Delete organization confirmation

D Dataset Ownership Transfer

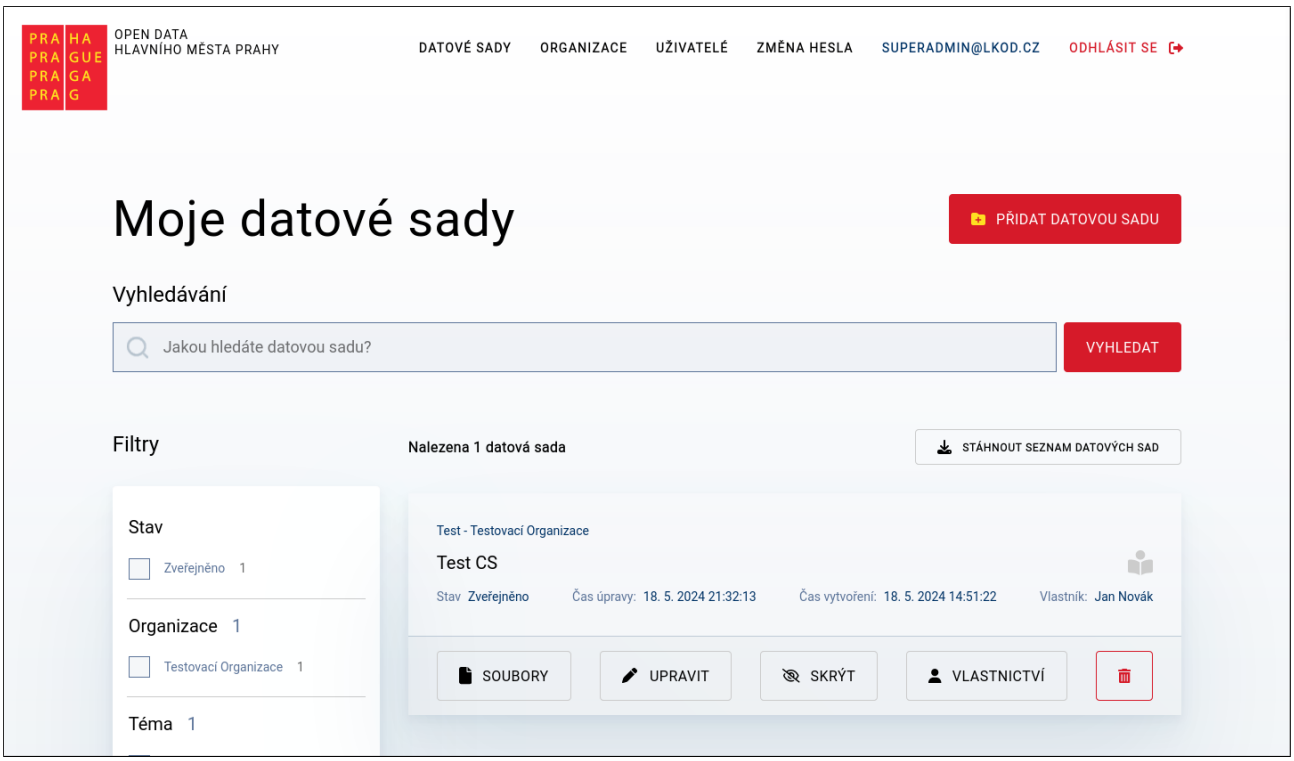

Figure D.1: Updated organization list view

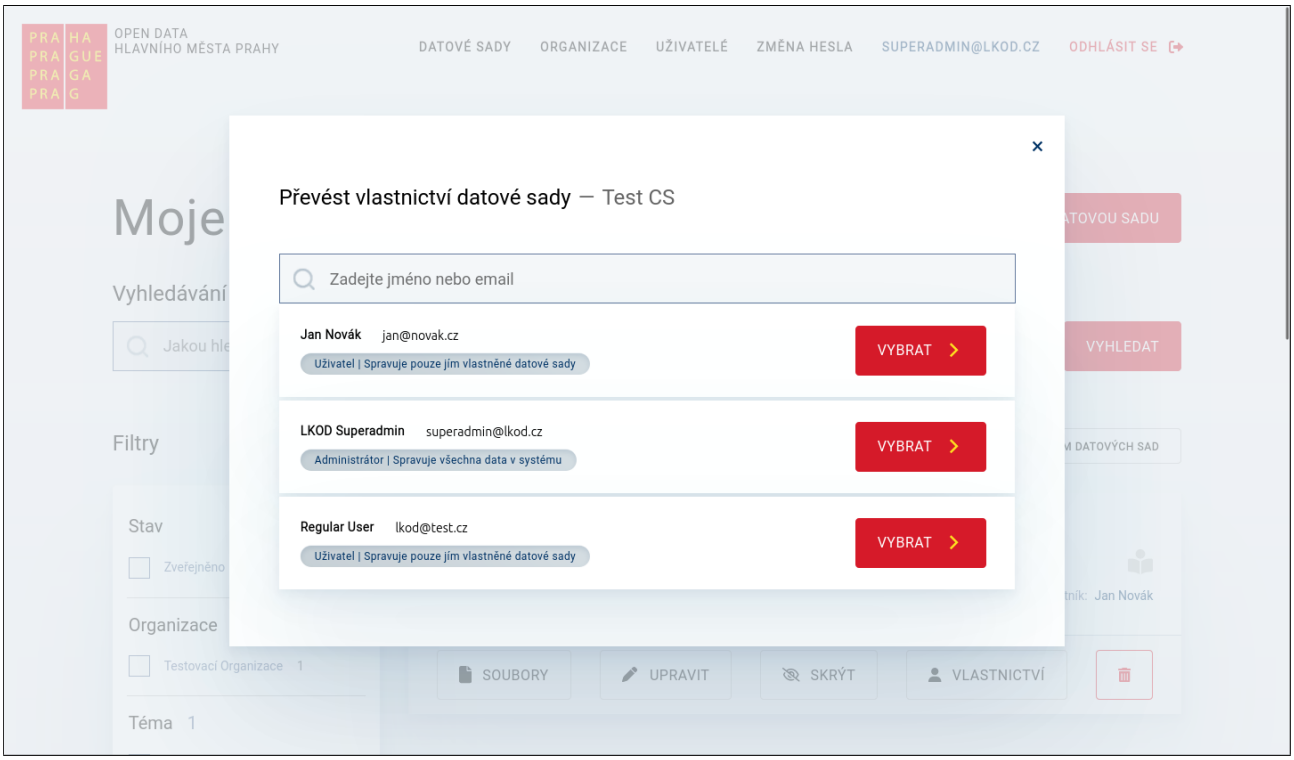

Figure D.2: Transfer organization ownership dialog

E Hradec Králové customization - Administration

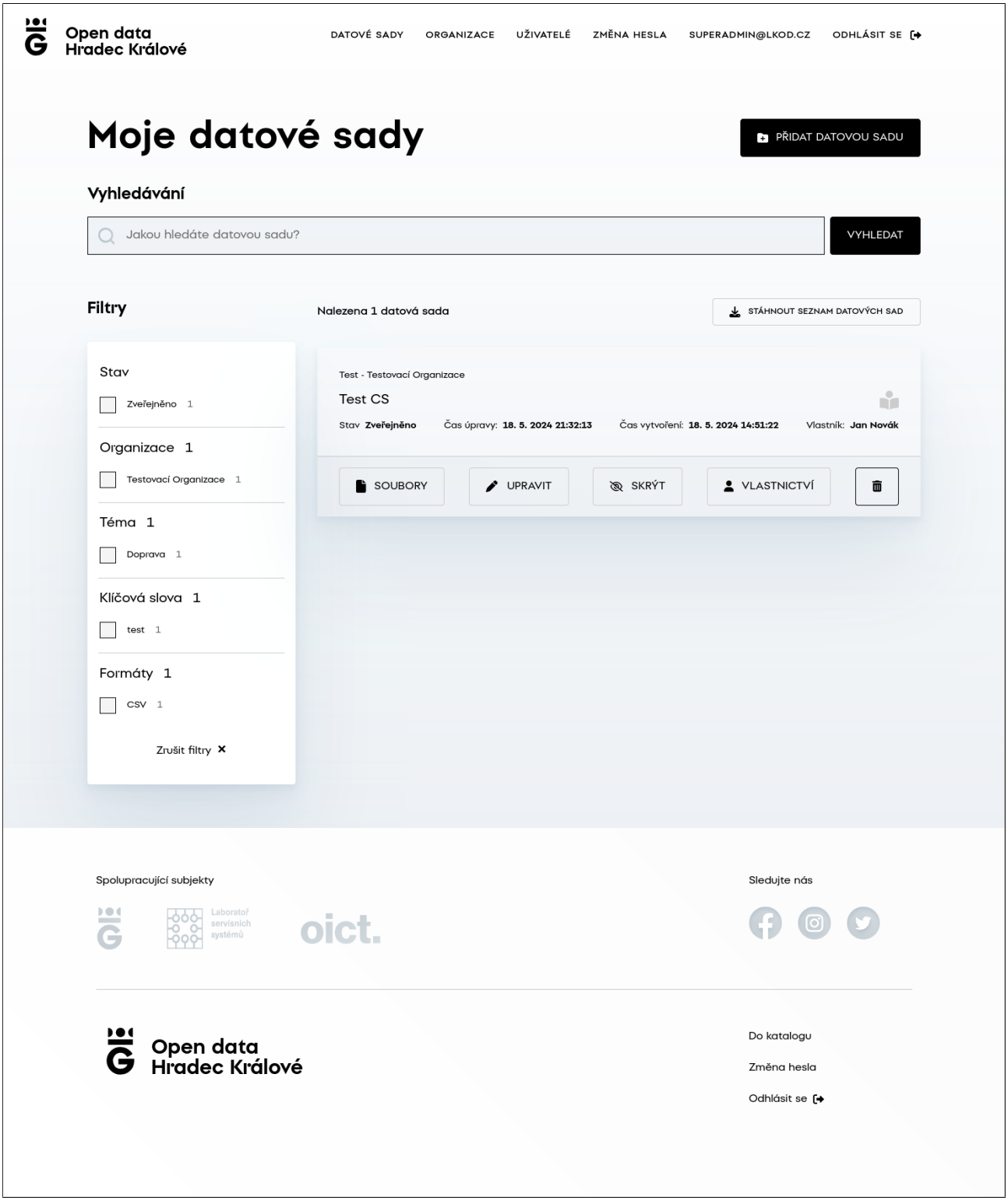

Figure E.1: Administration - Hradec Králové style

F Hradec Králové customization - Catalog

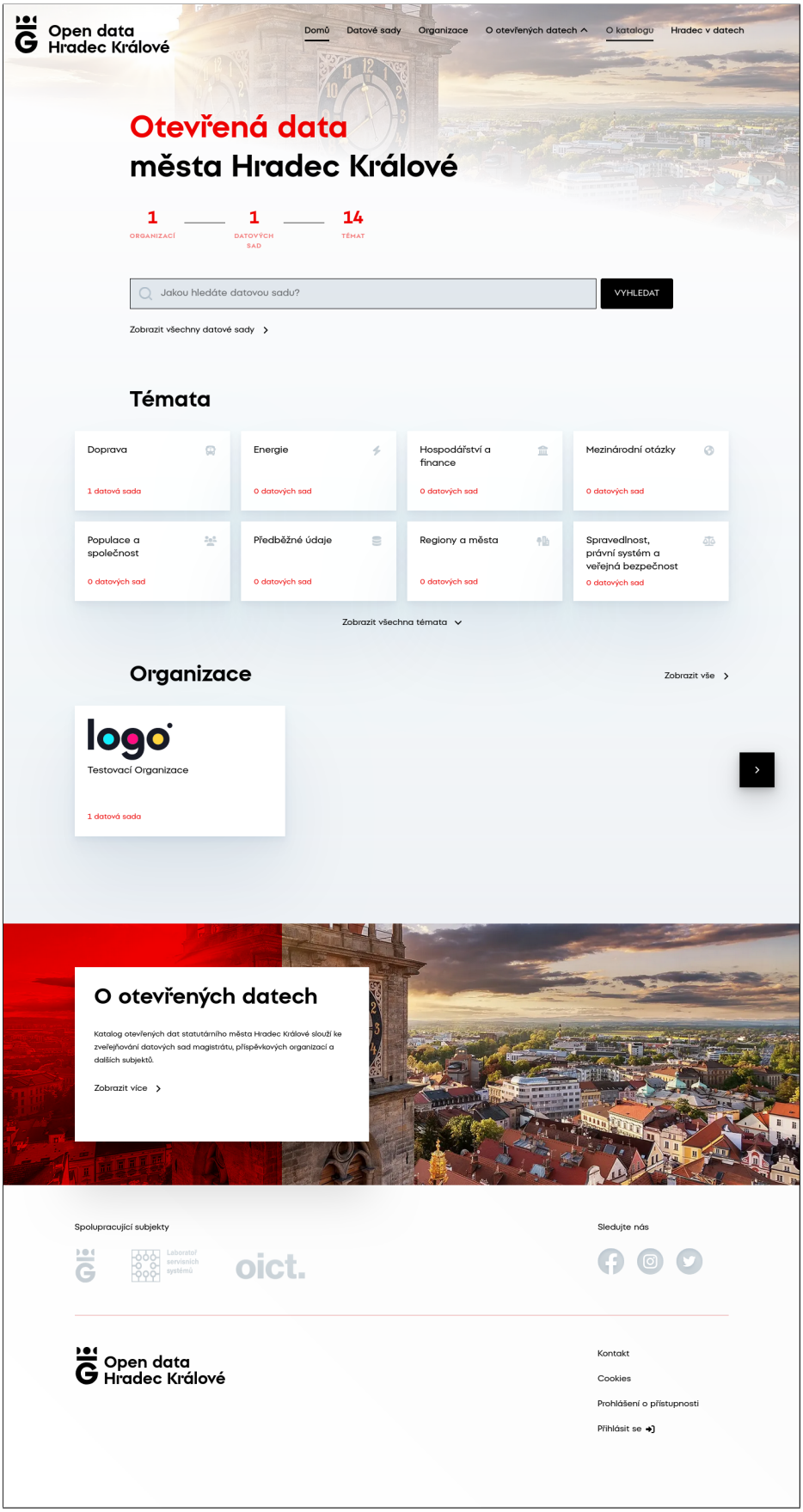

Figure F.1: Catalog - Hradec Králové style - Home

Figure F.2: Catalog - Hradec Králové style - Use and benefit of open data

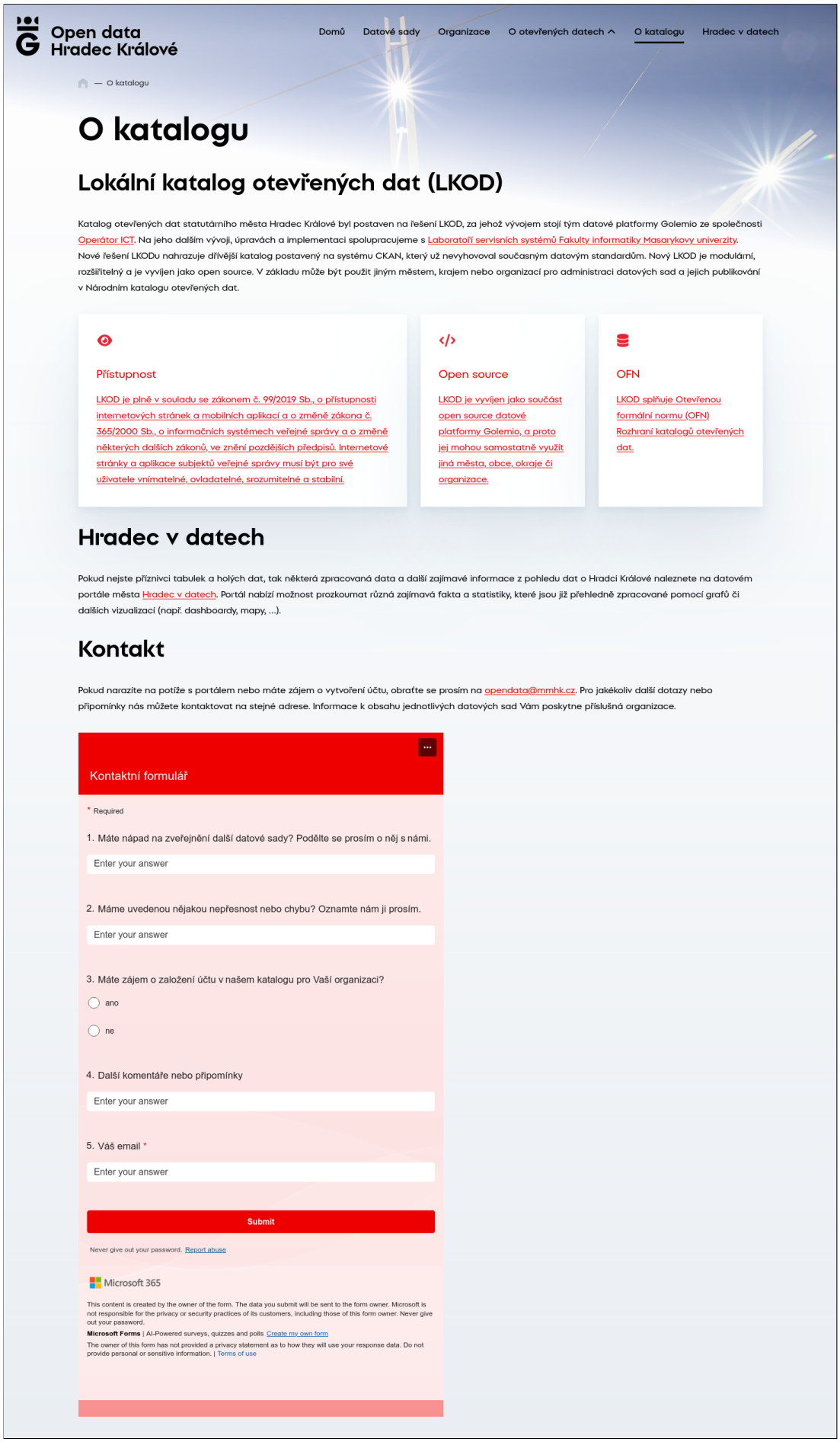

Figure F.3: Catalog - Hradec Králové style - About the catalog

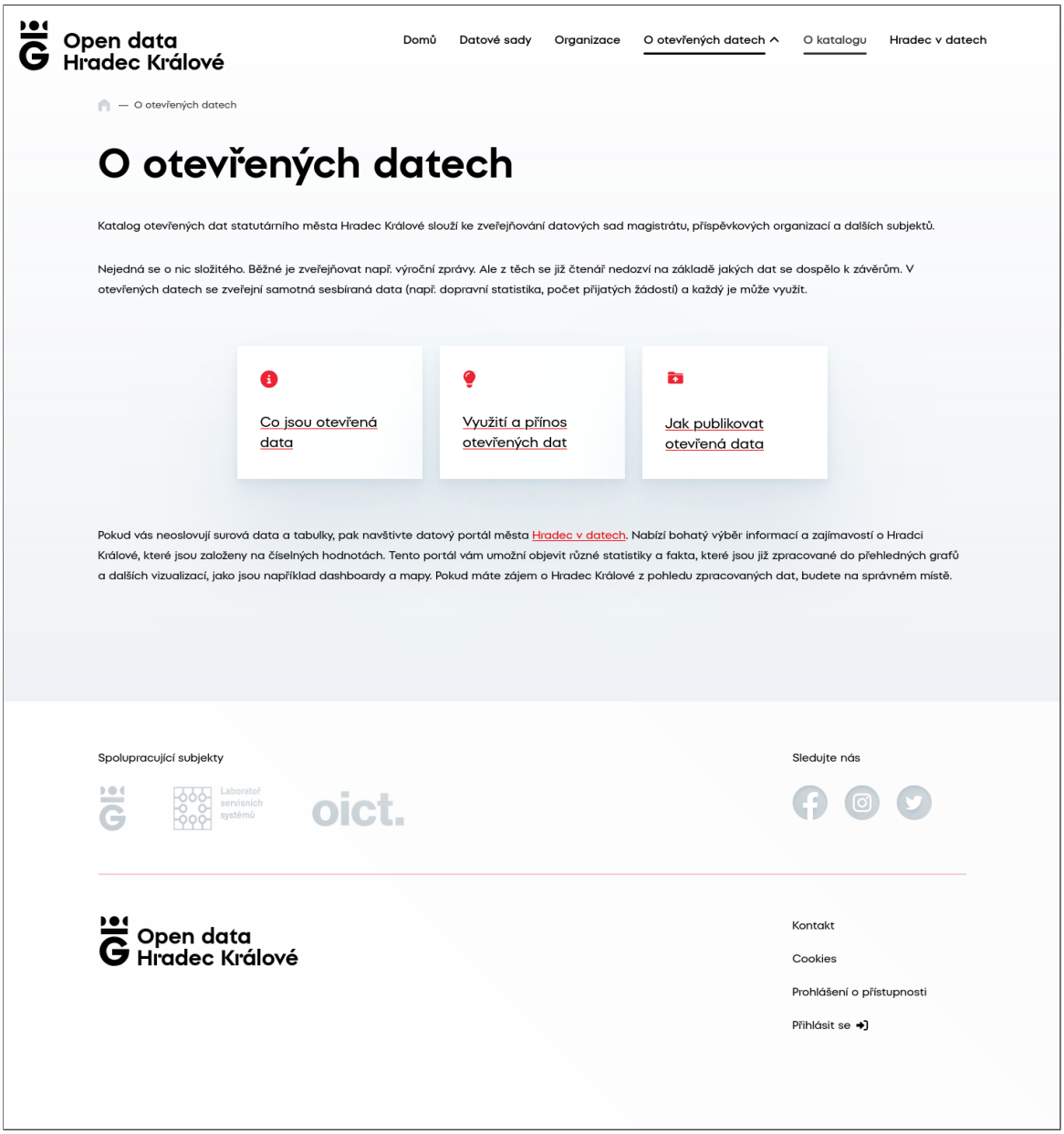

Figure F.4: Catalog - Hradec Králové style - About open data

Figure F.5: Catalog - Hradec Králové style - What is open data

G Source code

The source code of LKOD and CLM is attached to this thesis in the form of an electronic archive. Both projects include the modifications implemented as a part of this thesis, with descriptions of the changes included in the changelogs of the corresponding LKOD components.

The LKOD project is attached twice, first with the Hradec Králové customizations, and the second with the default Prague visual style with no customization.

The directory structure is as follows:

- **lkod-hk**: LKOD with the Hradec Králové visual style
- **lkod-prague**: LKOD with the default Prague visual style
- **CLM**: The CLM application