MASARYK
UNIVERSITY

FACULTY OF INFORMATICS

# Web-based Explorer of Digitized Maps

Bachelor's Thesis

MARTIN SELECKÝ

Brno, Spring 2026

MASARYK UNIVERSITY

FACULTY OF INFORMATICS

# Web-based Explorer of Digitized Maps

Bachelor's Thesis

MARTIN SELECKÝ

Advisor: doc. RNDr. Barbora Kozlíková, Ph.D.

Brno, Spring 2026

FACULTAS ARTIS INFORMATICÆ • UNIVERSITAS MASARYKIANA •

# Declaration

Hereby I declare that this thesis 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.

During the preparation of this thesis, I used the following AI tools:

- Grammarly for grammar check,
- GitHub Copilot for faster code writing,
- Claude to improve my writing style.

I declare that I used these tools in accordance with the principles of academic integrity. I checked the content and took full responsibility for it.

Martin Selecký

**Advisor:** doc. RNDr. Barbora Kozlíková, Ph.D.

## Acknowledgements

I would like to thank my advisor, doc. RNDr. Barbora Kozlíková, Ph.D., for her guidance, patience, and valuable feedback throughout the work on this thesis. I am also grateful to doc. Mgr. Bc. Zdeněk Stachoň, Ph.D. and the researchers from the Department of Geography, Faculty of Science, Masaryk University, for their consultations, their feedback on the prototypes, and their willingness to test the upgraded application.

Finally, I would like to thank my family and friends for their continuous support during my studies.

# Abstract

This thesis focuses on redesigning the Map Collection website of the Department of Geography at the Faculty of Science of Masaryk University. The thesis is composed of a current state analysis, the analysis of new requirements for the website, and the implementation of the redesign. The new design aims to implement the University's corporate design onto the website as well as improve the website's user experience. Figma was used as a technology in the design of the new visual style; Typescript and React are used in the implementation of the website. Hosting of the former and the new implementation is and will be handled by the Institute of Computer Science of Masaryk University.

# Keywords

web application, digitized maps, React, Next.js, responsive design

# Contents

# List of Figures

# Introduction

Digital map collections are a specialized type of information system. Unlike library catalogues or archival databases, these collections display data in a visual format. When someone searches for a historical map of a specific region, they mostly do not rely on a document identifier where the map is stored or a keyword that describes the map. Instead, they define a geographic area, set a time range, and evaluate the query results based on how well each map covers the area and time of interest. The frontend of such an application that enables these types of searches is, therefore, the part that decides whether the user finds what they were looking for. It influences if and how quickly the user can find relevant maps, and whether they can be accessed on all their devices.

The application in focus of this thesis is the "Mapová sbírka Geografického ústavu PřF MUNI", the map collection of the Department of Geography, Faculty of Science, Masaryk University. It provides the core functionality that a system of this type requires: spatial search using a map-based rectangle, filtering by time period and scale, result sorting by three spatial criteria (area similarity, area coverage, and distance between centers of the areas), and access to individual map records in a form of their digitized scans. The interface is organized into five sections *Search, Maps, Globes, Cataloguing,* and information *About the Collection*.

The application is still functional and has served the department and its users for many years. Over time, however, several limitations regarding the used technologies and user experience appeared. The technology stack had grown outdated, which made the codebase harder to maintain and extend. The interface was designed exclusively for desktop screens and provided no responsive layout for mobile devices. The map interaction model required a keyboard modifier to move the search rectangle, which is incompatible with touchscreen input. The map records overview table offered no text-based filtering and no way to sort records by publication year. These limitations motivated the upgrade of the system, presented in this thesis. The request for the new version of the system came directly from the researchers at the Department of Geography, as it started to be a pressing need to update the current solution.

The main goal of this thesis is thus to modernize the frontend of the application while preserving its existing functionality. This means upgrading all frontend dependencies to current versions, redesigning the interface to work correctly on both desktop and mobile devices, improving the map interaction model, extending the filtering and sorting options on the records overview page, and aligning the user Interface with the Masaryk University corporate design. The work is limited to the frontend layer and does not include changes to the backend services, the data infrastructure, or the content of the collection.

The thesis is structured as follows. Chapter 1 provides an analysis and description of the original application. Chapter 2 explores similar solutions and compares them to the one being presented by this thesis. Chapter 3 defines the requirements for the upgraded version, derived from the iterative discussions with the researchers from the Department of Geography. Chapter 4 focuses on the technologies used, and Chapters 5 and 6 cover the design and implementation. Chapter 7 summarizes the feedback received from a set of researchers from the Department of Geography, and Chapter 8 contains the thesis Conclusion.

# 1 Analysis of the Current Solution

This chapter describes the original application available at `mapy.geogr.muni.cz`. Understanding the main concepts and functionalities of the website directly shaped the requirements defined in Chapter 3. This chapter covers the structure and content of the original website, its visual design and layout, its existing functionality, and the specific limitations that motivated the upgrade, forming the core content of this thesis.

## 1.1 Structure and Content of the Original Website

The original web application for the Department of Geography map collection is accessible at mapy.geogr.muni.cz. It serves as a digital catalogue and browsing interface for a collection of historical maps and globes connected to the TEMAP project [1]. The application contains five separate pages, which are accessible from the main navigation bar: *Search, Maps, Globes, Cataloguing,* and *About the Collection.* The *Search* section, visible in Figure 1.1, is the main landing page, where users interact with a standard interactive web map component and retrieve records from the collection. The *Maps* section provides access to individual map records and their digitized scans. The *Globes* section contains four historical globe scans and information about them. The *Cataloguing* section documents the TEMAP project methodology, and the *About the Collection* section provides general information about the collection and its institutional context.

## 1.2 Visual Design and Layout

The application uses a vertically arranged single-column layout. A decorative background image of a historical map establishes the visual character of the page. The navigation bar is positioned below the header, presents links to all five above-described sections as a horizontal row of text elements, and includes the institutional branding of Masaryk University and the title of the map collection on its sides. The main content area occupies the remainder of the page and is

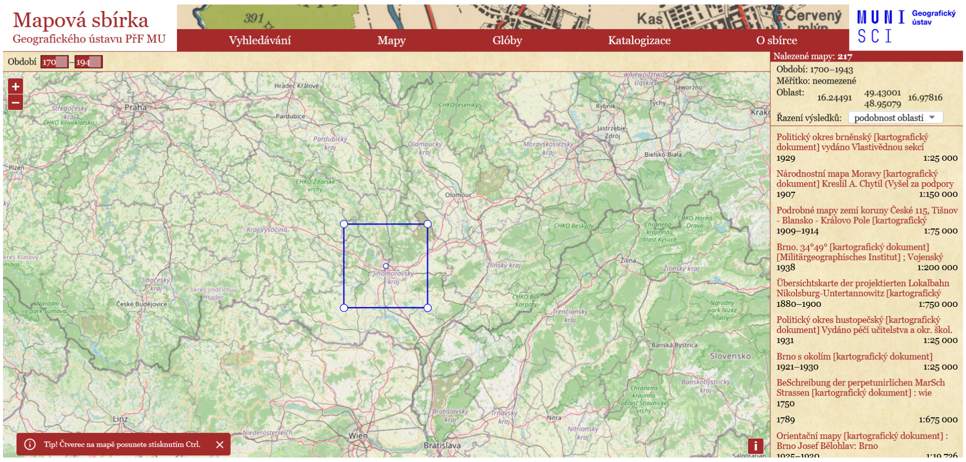


**Figure 1.1:** Landing page of the original application.

populated dynamically after the page loads via client-side JavaScript. The interface is available exclusively in the Czech language, which is another important limitation of the current version.

## 1.3 Existing Functionality

The *Search* section displays an interactive map that, by default, is centered on the whole Czech Republic. A rectangular selection area is drawn on the map, and the application retrieves a list of historical map records whose geographic extent intersects with this selection area. The resulting collection of maps is displayed as a scrollable list alongside the map and can be sorted by three criteria: similarity of area between the search rectangle and a found map, distance between centers of the search rectangle and a found map, and area coverage of the search rectangle by the found map area. Each criterion corresponds to a different spatial relationship between the search rectangle and the bounds of the map record. The results can also be filtered by a time range, which is set to 1700–1943 by default.

The *Maps* section presents either an overview table of all records in the collection or the full details of a single record. The detail view includes the record's metadata, such as the title, author, publication year, and scale, as well as access to digitized image scans displayed through a Zoomify tiled image viewer. The *Globes* section lists four historical specimens: the 1953 school physical-geographic globe, the 1940 Adler Erdglobus, the 1972 tectonic globe, and the Felkl globe from the early twentieth century. Each entry includes descriptive metadata and a digitized projection of the globe displayed in a two-dimensional map viewer. The viewer includes a button that switches the display to an interactive three-dimensional model of the globe.

## 1.4 UI Responsiveness

The main limitation of the UI of the current application is the responsive design. Since the application was primarily developed for desktop, it contains mostly fixed-width components, which are not displayed correctly when viewed on mobile devices. Navigation links become too closely spaced for reliable touch activation, and the layout does not adapt well to narrow screens.

The area search model, too, has a similar problem. Moving the search rectangle requires the user to hold the Ctrl key while dragging it. A hint in the bottom left corner of the map communicates this to the user. However, the interaction remains incompatible with touchscreen devices, as keyboard modifiers cannot be combined with touch input.

The application also provides no mechanism for contextualizing a map record within the larger series it belongs to. Many records in the collection are individual sheets from systematic multi-sheet map series. The original application presents each sheet as a separate entry with no visual indication of its position within the complete series grid, which sheets from the same series are held in the collection, and which positions remain undigitized. However, addressing this limitation, would require the creation of a new API service in the MUNI library information system. This falls outside the scope of this thesis.

Finally, the overview table of map records offers no text-based filtering. Users cannot search by author, title, scale, or geographic area. The only available filter is the time range, and the only sorting options are the three spatial criteria tied to the map rectangle. Records cannot be sorted by publication year.

# 2 Exploration of Similar Solutions

Before presenting the design and implementation of the upgraded application, it is useful to examine and get inspiration from platforms that address similar problems. The six systems analyzed in this chapter were selected because each one overlaps with the MUNI map collection application in terms of purpose, technical approach, or design. All six deal with browsing historical map collections, providing geographic or time scale search, or displaying digitized cartographic materials. Comparisons are made to the predecessor application at `mapy.geogr.muni.cz`, which the work in this thesis replaces and modernizes. Screenshots of each solution's interface are included as attachments.

## 2.1 OldMapsOnline

OldMapsOnline [2] is an open-access aggregation portal developed by Klokan Technologies. It indexes over 500 000 historical maps from more than fifty libraries and archives worldwide, including the British Library, Harvard Library, and the David Rumsey Collection. The main component through which the user interacts with the collections is a geographic sidebar search. As the user pans and zooms an interactive map, a results panel updates to list historical maps whose extents intersect the current viewport. A timeline slider narrows results by publication date, and a ranking algorithm orders them by geographic relevance.

The MUNI map collection application uses a similar approach. It differs by having a mode in which the user places a draggable square on the map and resizes it to define the search area more precisely. OldMapsOnline aggregates content from dozens of external institutions. The MUNI map collection application, by contrast, serves a single collection of approximately 18 000 items [3]. This makes it possible to include physical data alongside bibliographic metadata, such as barcode identifiers for individual physical copies. Aggregation portals cannot provide this level of institutional detail. There is also a technical connection between the two systems: Klokan Technologies developed the first OpenLayers-based Zoomify viewer, which is the same combination the MUNI map collection application uses for high-resolution scan display.

## 2.2 USGS TopoView

USGS TopoView [4] provides access to over 178 000 digitized historical topographic maps produced by the United States Geological Survey between 1884 and 2006. The layout is similar to the MUNI map collection application. The main content is an interactive map, and a results sidebar lists matching records that can be filtered by date range and scale. Selecting a result displays a scan preview and offers file downloads in GeoTIFF, KMZ, and JPEG formats.

TopoView is built around a single national map series, and its filters reflect this. The MUNI map collection covers a broader range of map types and supports three sorting types for results: percentage area similarity, area coverage, and distance from center. TopoView offers file downloads in GeoTIFF, KMZ, and JPEG formats, which require users to open files locally. The MUNI map collection instead displays scans directly in the browser via the Zoomify protocol, which better suits students and general visitors.

## 2.3 NLS Maps Finder

The National Library of Scotland's Maps Finder [5] is one of the largest public map portals in Europe. It holds material from the seventeenth century onward and shows the geographic extent of each available map as a coloured outline on an interactive base map. Users can identify and select items by location without typing any search query. The portal also offers advanced viewing modes: a side-by-side comparison viewer, a spyglass overlay, and a full-screen georeferenced viewer.

The MUNI map collection covers a narrower collection but is more closely integrated with its institution's services. The NLS portal does not link to library catalogue entries, does not show physical holding data, and does not allow filtering by scale. It also has no globe visualization. The MUNI map collection includes a *Globes* section that renders four historical globes from the MUNI collection using CesiumJS. Each globe can be viewed as a two-dimensional flat projection or switched to an interactive three-dimensional model.

## 2.4 David Rumsey Map Collection

The David Rumsey Map Collection [6] is one of the largest privately held digitized map archives in the world, with over 147 000 maps spanning the sixteenth to twenty-first centuries. It is the platform with the most overlap with the MUNI map collection. Both use a Zoomify-based viewer for scan display and CesiumJS for three-dimensional globe rendering. The David Rumsey site also includes a MapRank geographic search tool, an AI-assisted natural language search interface, and a Georeferencer overlay tool.

The collection is built for a global archive. Its viewing tools, geographic search, and georeferencing features are spread across multiple separate applications, which can make the interface difficult to navigate for new users. The MUNI map collection brings geographic search, scan viewing, globe rendering, and map sheet visualization together in one interface with a single navigation structure, but lack advanced filtration, context to map's content placement, and a responsive layout.

## 2.5 Czech Historical Atlas

The Czech Historical Atlas [7] was developed at the Czech Technical University in Prague. It presents 162 maps centered around Czech and Czechoslovak history. The topics that are included are primarily borders and territories, populations, and war conflicts. Each application supports time sliders, layer toggling, and clickable feature popups.

The two projects have different purposes. The Czech Historical Atlas uses maps to present historical arguments. The MUNI map collection gives users access to physical objects in a collection: scanned maps, historical globes, and their metadata. The goal is collection discovery, not historical analysis. The atlas has no catalogue integration and no spatial extent search. Both projects aim to make the Czech cartographic heritage accessible online, but they approach this from different directions. The atlas presents historical meaning through maps. The MUNI map collection provides access to the maps themselves as objects.

## 2.6 MapTiler Georeferencer

MapTiler Georeferencer [8] is a commercial tool for assigning geographic coordinates to scanned map images. Users place control points on the scan and on a reference base map shown side by side. The tool then computes a spatial transformation from those points. It includes multiple formats of output and has an interface spanning the whole viewport with toolbar controls placed directly over the map.

Georeferencer is an editing tool, not a collection browser. However, it is still relevant to the MUNI map collection. Firstly, its interface approach, a toolbar over a map spanning the whole viewport with dialogs for detailed operations, is comparable in design to the predecessor application's Zoomify scan viewer dialog. Second, the TEMAP initiative documented in the current application's *Cataloging* section used a closely related georeferencing workflow for Czech memory institutions. This connects the thesis work to the broader context that tools like Georeferencer represent.

# 3 Requirements for the Website

A definition of requirements that are expected from a project is a crucial step in web development. Since the definition represents a mutually agreed specification of the project, or in this case, project modifications and additions, between the client and the developer, it is imperative that both sides find the definition clear and precise. Furthermore, to prevent any delays to the project's completion, the definition should be altered as little as possible.

The requirements for the website implemented in this thesis were discussed and agreed upon during consultations with the thesis advisor and thesis consultant. The elementary requirements of the website were already defined and implemented in the previous version of the website, so the requirements defined for this thesis project were primarily focused on changes that were requested by the researchers from the Department of Geography and were mostly related to the UI and the addition of new functionality.

## 3.1 Responsive Layout and Mobile Usability

The first category of requirements concerns the responsiveness of the user interface. The original implementation of the website was designed exclusively for desktop screens with widths of approximately 1200 pixels or more, which caused fixed width containers to overflow the viewport on smartphones and tablets, navigation links to become too closely spaced for reliable touch activation, and the map interaction model, which required the user to hold the Ctrl key while dragging the search rectangle, to be incompatible with touchscreen input. The new implementation is therefore required to deliver a fully responsive layout that functions on desktop as well as on mobile devices. The navigation bar must transform at narrow viewport widths from a horizontal row of text buttons into a compact hamburger icon that opens a side drawer, and the homepage must adapt its layout so that on mobile screens the filter panel is presented as a swipeable bottom drawer following the interaction pattern established by popular consumer mapping applications. All interactive elements, including the handles used to manipulate the search, must be sized large enough to be operated comfortably with a finger on a touchscreen without the risk of accidentally activating an unintended control, and the map interaction model must be redesigned to operate without keyboard modifiers on both desktop and touch devices.

## 3.2 Multilingual Support

The second category of requirements concerns the addition of a complete English translation of the user interface. The original system offered no language choice; every element of the interface was in Czech, which constrained the platform's reach to a narrow audience. The new implementation is required to maintain a dictionary that includes translation of every user-visible string, has both a Czech and an English entry, and to provide a language toggle button in the application header that switches between the two at any moment during a session, with the switch taking effect immediately across all visible components without a page reload. The translation coverage must be comprehensive and must include navigation labels, page headings, metadata field names, sort mode labels, time filter controls, table column headers, pagination controls, error and loading messages, and all informational text on the *Cataloging* and *Globes* pages. The system must also be designed to make the future addition of further languages straightforward.

## 3.3 Technology Upgrades and Visual Redesign

The third category of requirements is technical and architectural. The entire technology stack of the original system, including TypeScript, Redux Toolkit, Material UI, React, Next.js, and OpenLayers, was already in place but carried over in outdated versions that limited the ability to introduce new functionality cleanly. Upgrade requirement, therefore, requires upgrading each of the dependencies to the current newest long-term support major version. Ensuring that the dependencies can be used to develop without breaking any functionality of the web application. CSS Modules, which were already used for component styling to prevent class name collisions, are to be retained and applied consistently across any newly added components. Together with the dependency upgrades, the visual style of the application needs to be altered to follow the official Masaryk University corporate design, replacing the decorative antique map background with a clean layout using the university's defined color palette and typography. The map record dataset must continue to be fetched once per user session from the ArcGIS Feature Service and stored in the Redux state, with all filtering operations computed entirely on the cached in-memory dataset.

## 3.4 Filtering, Sorting, and Data Infrastructure

The final requirement concerns the extension of the existing filtering and sorting functionality available on the map records overview page. The transformation must not represent a reduction or simplification of what the platform offers. Every feature that users of the original system relied upon must be preserved or reproduced in the upgraded system. In addition, the map records overview table must be extended to allow sorting records by publication year in both ascending and descending order, in addition to the existing spatial sorting modes. Individual text fields that allow the user to search by longitude, latitude, author name, map title, and scale must be added to the filtering functionalities. The transformation is required to remain a pure frontend upgrade, leaving the underlying data infrastructure entirely unchanged: the ArcGIS Feature Services at `maps.muni.cz`, the Zoomify tile server at `mapy.geogr.muni.cz`, and the WMS endpoints at the Czech cadastral authority and at Masaryk University's own ArcGIS instance must continue to operate as before, with the new application acting as a client-side presentation layer that consumes the same API endpoints as the original system.

# 4 Technologies Used

The upgraded version of the geographic map collection web application builds on the technology stack of the original implementation. The aim of the thesis was not to create a completely new system, but to modernize an existing web application used as a digital catalogue and browsing interface for historical Czech geographic maps and globes. The application is connected to the TEMAP project and is developed in the context of the Department of Geography, Faculty of Science, Masaryk University.

The backend data infrastructure remained unchanged. The application still uses existing ArcGIS REST Feature Services and the Zoomify tile server hosted on the university map infrastructure. Therefore, the main technological focus of the thesis is the frontend layer. Most of the technologies described in this chapter were already present in the original implementation. However, they were outdated and had to be rewritten from scratch or upgraded to newer versions.

For this reason, this chapter not only describes the technologies themselves. It also explains why they remained suitable for the project after the upgrade. The selected technologies support the main goals of the thesis: better maintainability, improved usability, visual consistency, responsiveness, and long-term sustainability. Specific implementation details, such as component structure, routing, state slices, and map configuration, are described later in the implementation chapter.

## 4.1 React, Next.js, and TypeScript

The core of the frontend is built on three closely related technologies: React, Next.js, and TypeScript. All three were present in the original implementation and were upgraded as part of the modernization process.

React is a JavaScript library that uses reusable components to build user interfaces [9]. This model is well-suited to the application, as the interface consists of elements that appear across multiple pages: navigation bars, filtering controls, map panels, record tables, and dialogs. The component structure made it possible to modernize individual parts of the interface without a full rewrite, which was important given that the backend services were preserved and only the frontend was changed.

Next.js is a React framework that provides file-system routing, server-side rendering, and code splitting [10]. The routing model maps directly onto the page structure of the application, and code splitting ensures that large libraries are only loaded on the pages that need them [11]. This is particularly relevant for the *Globes* page, which depends on the Cesium.js library, a large 3D rendering engine that would otherwise increase the loading time of every page in the application.

TypeScript extends JavaScript with static typing [12]. Because the application processes structured data from external geographic services, explicit type definitions make it clear what type of data will be received. This means that mismatches between the API response format and the internal data model are caught before the application runs in the browser rather than after. TypeScript does not affect the visual output of the application directly, but it reduces the risk of errors during development and makes the codebase easier to navigate.

## 4.2 Redux Toolkit

Redux Toolkit was used for global state management. It is the recommended toolset for Redux logic and reduces common problems such as excessive boilerplate code and complex store configuration [13]. As with the other dependencies, it was already present in the original application and was upgraded as part of the same process.

The application needs several parts to access the same data and read or update their state. This includes:

- loaded map records,
- active filters and sorting settings,
- selected map data,
- language settings, and
- viewer-related state.

Because these values are used across different pages and components, they cannot be managed in isolated local states. Redux Toolkit keeps them in a single storage and provides a clear structure for actions, reducers, and selectors. This separates the application logic from the visual components, so each component focuses on presentation rather than data handling. The exact structure of the store is described in the implementation chapter.

## 4.3 Material UI and CSS Modules

Material UI is a React component library based on Google's Material Design system [14]. It provides prebuilt interface elements such as navigation bars, buttons, forms, tables, dialogs, drawers, sliders, and layout containers. Its use in this project was not limited to individual components. Material UI also allows colors, typography, and spacing to be defined once and applied consistently across the entire interface [15]. This made it possible to align the visual style with the university's corporate design without repeatedly overriding the default design.

CSS Modules handle the style of individual components that have to be styled differently from the general CSS style. Each component has its own `module.css` file, and the class names defined in it have a local scope [16]. This prevents styles from one component from affecting the layout or appearance of another. In practice, Material UI covers global visual consistency, while CSS Modules handle custom layout rules and adjustments specific to individual components.

## 4.4 OpenLayers

OpenLayers is a JavaScript library for interactive web maps [17]. It supports working with data sources including WMS layers, vector geometry, tiled image formats, and geographic projections. In this application, it is mainly used to render the interactive search map on the homepage, where users move and adjust a rectangle to define a geographic search area. It also displays the sheet overview panel on the map record detail page, where vector rectangles mark the position of each sheet within a historical map series. Finally, it powers the scan viewer, where it renders high-resolution map scans delivered via the Zoomify tiled image protocol.

The decision to keep OpenLayers was mainly determined by the fact that the previous version of the application had implemented it and that it is compatible with the existing infrastructure. The application performs spatial calculations in the Mercator projection and then displays the coordinates to users, which OpenLayers supports without any data alteration needed. Furthermore, the university map services that the application depends on expose WMS endpoints, which OpenLayers supports directly. Therefore, replacing the OpenLayers library would have required building new map-related functionality from scratch with no clear benefit.

## 4.5 Figma and Git

Figma and Git supported the development process without becoming part of the deployed application.

Figma is a design tool for interface design and prototyping [18]. It was used during the design phase to plan the layout, typography, color scheme, and component placement of the upgraded interface before any code was written. Figma made it possible to test and adjust MUNI corporate design visuals, new interface layout, and mocking of the proposed functionality, without implementing the requirements, which reduced the number of significant layout changes needed later in development, due to further specification or adjustments of the requirements.

Git is a distributed version control system [19]. It was used to track all changes to the codebase throughout the project. It was particularly useful because the transformation involved both upgrading existing code and introducing new features at the same time. Git made it possible to work in smaller, isolated steps and to recover earlier states of the code when needed.

# 5 Design

The design phase preceded the actual implementation of the application. Before any code was written, the layout, color scheme, typography, and component placement of the upgraded interface were consulted with the researchers at the Geography Institute of the MUNI Science Faculty and planned in Figma. This step was particularly important because the upgraded application was required to follow the Masaryk University corporate design. Visual decisions about colors, spacing, and navigation structure had to be evaluated before they were reflected in the code. Revising those decisions once they are embedded in a component hierarchy is considerably more difficult than adjusting a frame in a design tool.

The Figma file contains two pages: Design and Components. The Design page holds frames for all five pages of the application, each designed to be 1440 pixels wide. Czech and English versions of every page are included, and several frames capture different states of the same page. For example, the *Search* page with the results panel active, and the same page with the filter panel open instead. The file containing the complete Figma design is included in the attachments of this thesis.

## 5.1 Visual Identity

The visual identity of the upgraded application was requested to be based on the official Masaryk University corporate design[1]. The original application used a decorative background image of a historical map as its primary visual element. The upgraded application replaces this with a clean layout that applies the university's defined color palette.

The primary color is SCI MU green, which appears as a solid strip across the navigation bar. This color is also used for the footer bar and as the secondary color for interactive controls. Content areas use a white background. Table column headers and secondary surfaces use light gray to create visual separation without adding weight to them.

1. https://sablony.muni.cz/weby

Black is used for all body text. The result, example of which is shown in Figure 5.1, is a high-contrast, neutral interface, implementing the official color palette of SCI MUNI, that keeps the focus on the map and globe content rather than the components around it.

All icons are drawn from the Material UI icon set, consistent with the component library used throughout the implementation. Navigation labels, column headers, filter labels, and button text all use a single sans-serif typeface throughout.

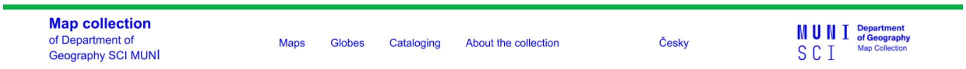


**Figure 5.1:** New interface design implemented in the navigation bar, following the SCI MUNI corporate design.

## 5.2 Navigation and Layout Structure

The navigation bar appears at the top of every page. It contains the collection title and faculty name on the left, navigation links to all four sections centered in the bar, a language toggle on the right, and the SCI MU Map Collection logo at the far right. The strip spans the full 1440-pixel width and uses the standard SCI MU green color.

The *Maps* page uses a split layout visible in Figure 5.2: a 500-pixel sidebar on the left and the interactive map filling the remaining 940 pixels on the right. The sidebar contains two tabs, Maps and Filters, that toggle between the results list and the filter controls. A thin arrow element on the right edge of the sidebar allows the user to collapse it entirely, leaving the map in full-width view.

## 5.3 Page Designs

The design contains four separate pages: *Maps, Globes, Cataloging, About the Collection.* The Maps tab in the sidebar displays map records as a scrollable list. Each entry shows the thumbnail placeholder, a record name, and a short description. A "Show all maps" button is pinned to the bottom of the sidebar and opens the all maps overview as a modal dialogue.

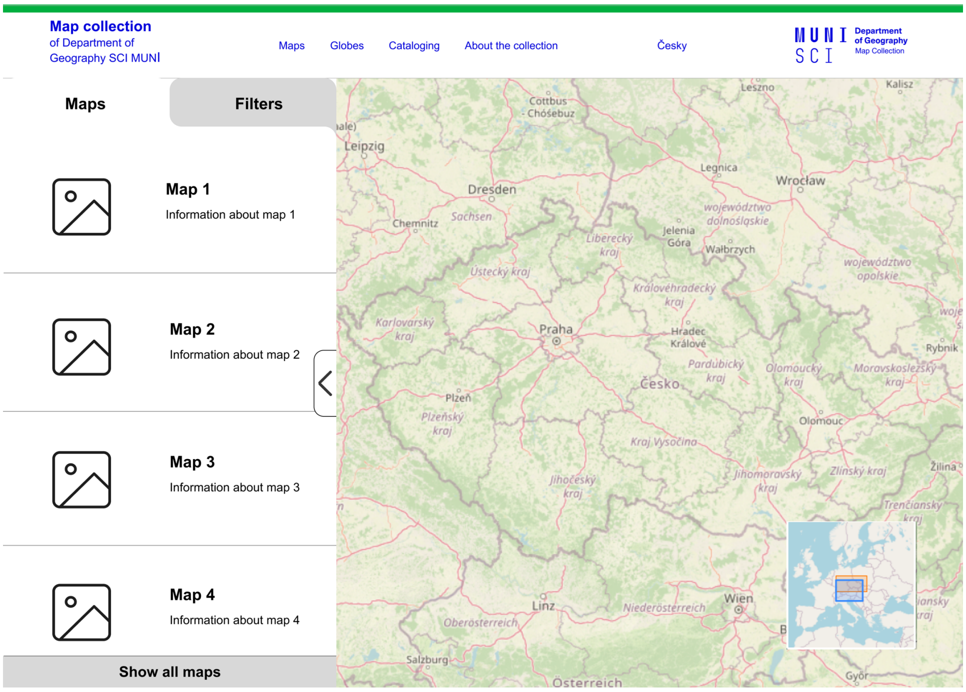


**Figure 5.2:** Figma screenshot of the search page design with the *Maps* tab active.

The Filters tab replaces the results list with filter controls. The Figma design organizes these into three groups: two checkbox groups and one date range slider. The slider runs from 0 to 2026 and corresponds to the time range filter already present in the original application, as is shown on the left side of Figure 5.3.

The all maps overview, visible in Figure 5.4, is designed as a dialogue window 1000 pixels wide and 800 pixels tall. It contains a table with five columns: Picture, Name, Author, Year published, and Scale. Each column header carries a sorting arrow. A filter icon in the top left of the dialogue opens the advanced filter dialogue. A close button in the top right dismisses the dialogue.

The advanced filter dialogue covers the same surface area. As shown in Figure 5.5, it organizes filters into four groups: a text search input, two range sliders for Scale and year, a two-column checkbox

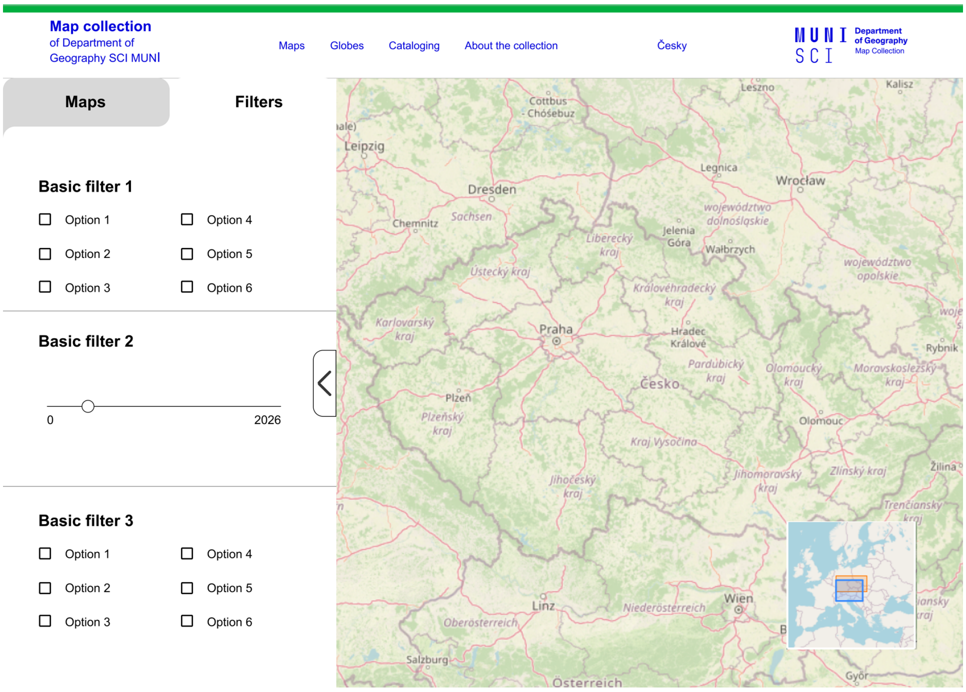


**Figure 5.3:** Figma screenshot of the search page design with the Filters tab active.

group, and a four-column checkbox group. This layout was designed to provide comprehensive filtering options in a single panel.

The *Globes page* uses a vertical single-column layout. Each of the four historical globes is presented as a named section with a photograph of the physical specimen alongside its metadata. *Cataloguing* and *About the Collection* pages use the same single-column layout, both primarily displaying text information about the map collection project, as shown in Figure 5.6.

## 5.4 Deviations from the Figma Design

Two structural differences exist between the Figma design and the delivered implementation. Both arose from practical decisions made during development.

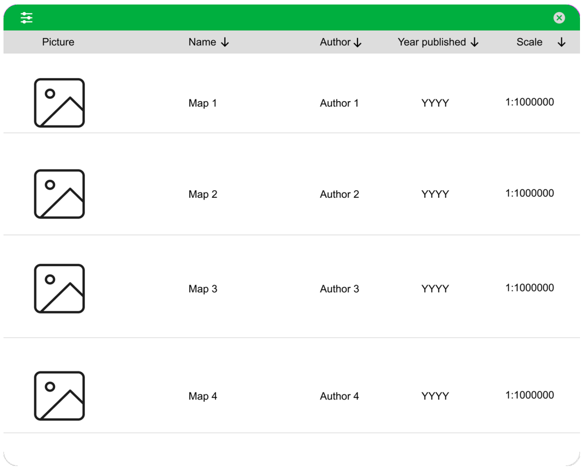


**Figure 5.4:** Figma screenshot of all maps dialog design.

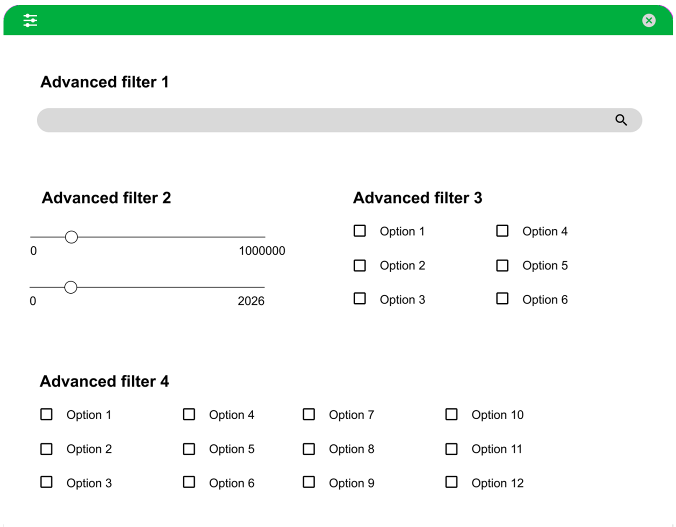


**Figure 5.5:** Figma screenshot of the advanced filter dialog design.

The first concerns the maps overview and filters. In Figma, browsing all records and applying advanced filters are modal dialogues that appear over the *Search* page. In the implementation, map browsing is a dedicated page at its own route, and the filter controls appear as a row

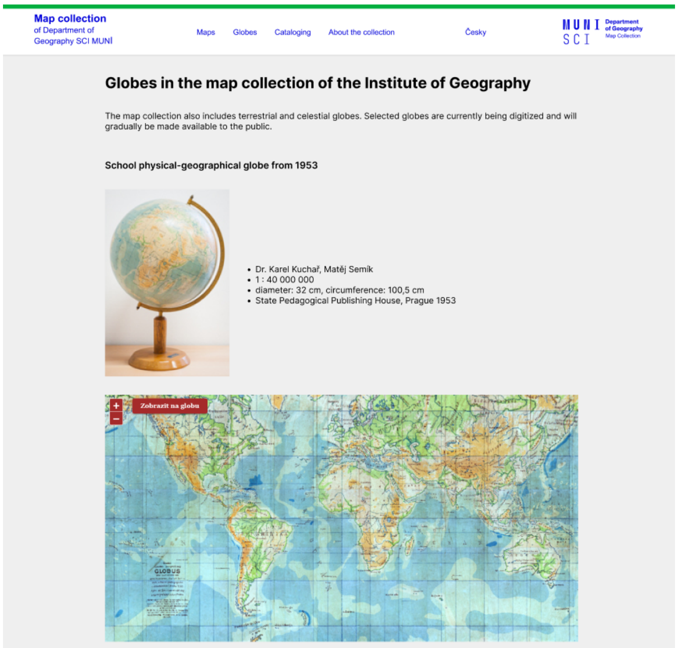


**Figure 5.6:** Screenshot of the Globes page design.

of text input fields directly above the table rather than inside a separate model. This change made the filtering interface more accessible and removed the need to manage modal state across the application.

The second concern is the Figma file's home screen frame, which shows a minimal landing page with two large cards linking to the *Maps* and *Globes* sections. This frame was an early concept, and after consultation with researchers at Geography Institute, it was decided the frame was unnecessary to the website's function, would confuse the user, and therefore was not carried into the implementation. The *Search* page serves as the landing page in the delivered application.

Minor visual differences also exist in spacing, padding values, and the exact behaviour of interactive elements such as sliders and drawers, where Material UI component defaults were followed rather than replicating the Figma layout precisely. These differences do not affect any functional requirement defined in Chapter 3 .

# 6 Implementation

The predecessor application at `mapy.geogr.muni.cz` was a functional project. However, as already stated, it had limited component-based state management, no responsive layout for mobile screens, no multi-language support, and a limited sorting and filtering functionality. This chapter describes the process of replacing the old codebase with an upgraded application that improves these shortcomings and follows the previously described design created in Figma.

The chapter covers the key areas of the implementation: dependency upgrades, UI remodel, interactive map controls, mobile layout, the search and filter system, the map records overview page, and the translation system. For each one, the chapter describes the changes made and the reasons behind them. The source code of the implementation of this thesis is included in the attachments.

## 6.1 Dependency Upgrades

The predecessor application used Next.js 14.0.4 [1], React 18[2], OpenLayers 8.2.0[3], and Redux Toolkit 2.0.1[4]. The upgraded application uses Next.js 16.2, React 19.2, OpenLayers 10.9.0, and Redux Toolkit 2.11.

The newer versions of these libraries improve the performance, API design, and TypeScript support that the new implementation relies on. Several of the features added in this project, including the custom map interactions, the extended Redux state, and the updated Material UI integration, require the newer versions to function correctly and would not be achievable on the original dependency versions.

The `headroom.js` library was added to support a navigation bar that hides on scroll down and reappears on scroll up. The `@fortawesome` icon packages were added to supply icons used in the header and sidebar.

1. https://nextjs.org/docs
2. https://react.dev/
3. https://openlayers.org/en/latest/apidoc/
4. https://redux-toolkit.js.org/

## 6.2 Visual Redesign and Theming

The design decisions outlined in Chapter 5 required concrete changes to the Material UI configuration and the specific component styles. The original theme defined the primary color as #a52a2a (a medium brown) and the font family as Georgia, a serif typeface. The main page also displayed a decorative background image of a historical map behind the header. These were replaced by the colors and typography of the Masaryk University corporate design: the primary color was set to #0000dc, the font family to Arial, and the background image was removed.

A green strip `border-top: #00af3f 5px solid` was added to the top of the header. This color corresponds to the Faculty of Science and distinguishes the website as a project of that faculty.

The AppBar (Figure 6.1) component's `elevation` property was set to 0, which removes the default Material UI drop shadow from the navigation bar. The original version included a raised shadow under the header, which was inconsistent with the flat layout used in the MUNI corporate design.

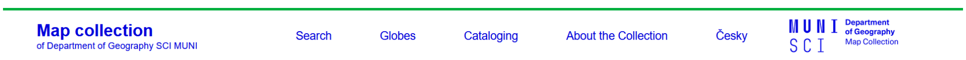


**Figure 6.1:** Screenshot of the application header styled in the new design.

## 6.3 Navigation, Language Switcher, and Footer

The navigation bar component was extended with two new elements: a language switcher and a footer. In the original application, all navigation labels were hard-coded Czech strings. The upgraded version replaces all hard-coded labels with calls to the translation function, so the labels change when the user switches the language.

The LanguageSwitcher component is a new addition to the header. On desktop screens, it is placed in the navigation bar alongside the existing page links. On mobile screens, a shorter version of the component is displayed. The switcher sends a *languageChanged* action to the Redux store when clicked.

The navigation items array was also changed. The original header included a direct link to the map records page (`/mr`) as a navigation item. This link was removed from the main navigation in the new version. The map records page is now accessed from a button at the bottom of the search results panel.

A footer component was added to all pages. The original application had no footer. The new footer contains a copyright notice with the current year, a link to the Masaryk University website, and a link to the Institute of Computer Science administrative page on the left side of the page. On the right side is a social media section that links to the Faculty of Science accounts on Facebook and Instagram using FontAwesome brand icons. Both the footer text and the social media links support the Czech and English translation system. The complete footer layout is shown in Figure 6.2.


**Figure 6.2:** Screenshot of the new footer.

## 6.4 Global State Extension

The Redux state in the original implementation tracked the map records dataset, the time range filter, the selected aggregation method, the rectangle extent, and a highlighted extent for hover effects. The upgraded implementation adds six new state fields: *language*, *scaleRange*, *titleFilter*, *authorFilter*, *searchMode*, and *viewportExtent*.

The language field stores the current UI language as either '`cs`' (Czech) or '`en`' (English), with '`cs`' being the default. The scaleRange field stores a range of integers that filters records by map scale. The titleFilter and authorFilter fields store text strings used for substring matching to the corresponding metadata fields of each record. The searchMode field toggles between '`rectangle`' mode, in which the drawn rectangle defines the search area, and '`viewport`' mode, in which the currently visible map area is used. The viewportExtent field tracks the current map view bounding box and is updated on every pan and zoom event.

The initial value of the time range lower bound was also changed. The original implementation initialized *timeRange* from 1500 to 1700. The new version sets it to 1500, which extends the search range to cover older maps in the collection that predate 1700.

Each new state field has a corresponding action creator. The new *scaleRangeChanged*, *titleFilterChanged*, *languageChanged*, *searchModeChanged*, *authorFilterChanged*, and *viewportExtentChanged* actions are generated by Redux Toolkit's `createSlice` and used directly in the relevant components.

## 6.5 Map Interaction Redesign

The original application used two OpenLayers interactions to manipulate the search rectangle: a Modify interaction for resizing and a Translate interaction for moving. The Modify interaction is a built-in OpenLayers class that provides handling points to a polygon. The Translate interaction is another built-in class that moves a selected object when the user drags it. The key limitation of this combination was that Translate captured all drag events on the feature, which meant the user had to hold the Ctrl key to distinguish moving from panning on the base map. This keyboard modifier made the interaction incompatible with touchscreen devices.

Both interactions were removed and replaced with two new custom interactions written from scratch. The first, `cornerResize` in `src/core/interactions/cornerResize.ts`, extends the OpenLayers Pointer class. On each pointer move event, it checks whether the pointer is within 16 pixels of any of the four corners of the rectangle. If so, it changes the CSS cursor to the diagonal resize cursor and marks the corner as the active target. On a pointer down event at a corner, a dragging begins. During the dragging, the interaction updates the two sides of the rectangle that the active corner belongs to while keeping the other two sides fixed, then redraws the rectangle. On pointer up, it saves the final form to the Redux store.

The Pointer is also extended by second interaction, `sideTranslate` in `src/core/interactions/sideTranslate.ts`. It activates when the pointer is within 10 pixels of any edge of the rectangle. A custom moving cursor is applied. During the dragging, the interaction calculates the coordinate difference between the current and the previous pointer position and updates all four sides of the rectangle by that difference, keeping the rectangle size constant. The final position is sent to the store on pointer up.

Because these interactions respond only to pointer events on a specific part of the rectangle and all other pointer positions resolve on the base map, no keyboard interaction is required. Figure 6.3 shows the selection rectangle in the context of the redesigned landing page and with the MiniMap overlay.

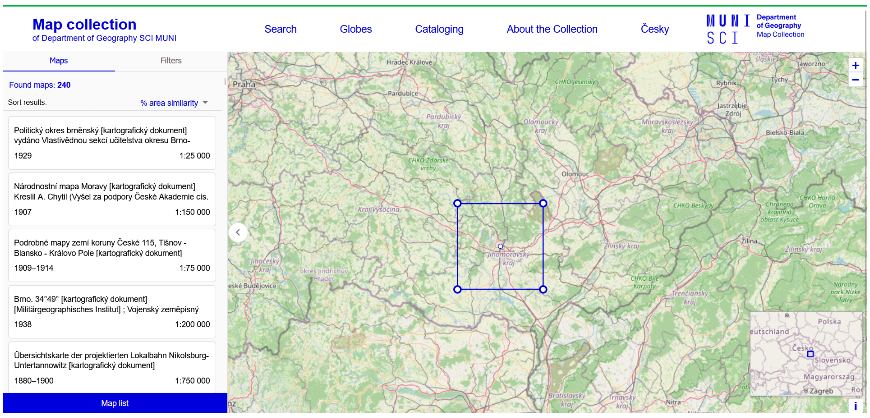


**Figure 6.3:** Screenshot of the redesigned homepage, selection rectangle, and the minimap context.

## 6.6 Homepage Layout and Viewport Search Mode

The homepage is composed of four elements: the application bar at the top, a sidebar panel on the left for desktop screens, a full-width interactive map in the remaining space, and a bottom drawer for mobile screens. The sidebar and the bottom drawer each contain a Panel component with the search results list and a FilterInfo component with the filter controls.

On map initialization, two event listeners are activated. The first starts on *rendercomplete* and sends the initial map area to the Redux store. The second starts on every *moveend* event and sends the updated extent. Each of the listeners is calling the function `map.getView().calculateExtent(map.getSize())` and send the result as a *viewportExtentChanged* action.

In the viewport mode, shown in Figure 6.4, the search area is always the current map view rather than the draggable rectangle. When the user switches to viewport mode, the rectangle layer is hidden, and the selector for the active search extent switches to *viewportExtent* instead of *rectangleExtent*.

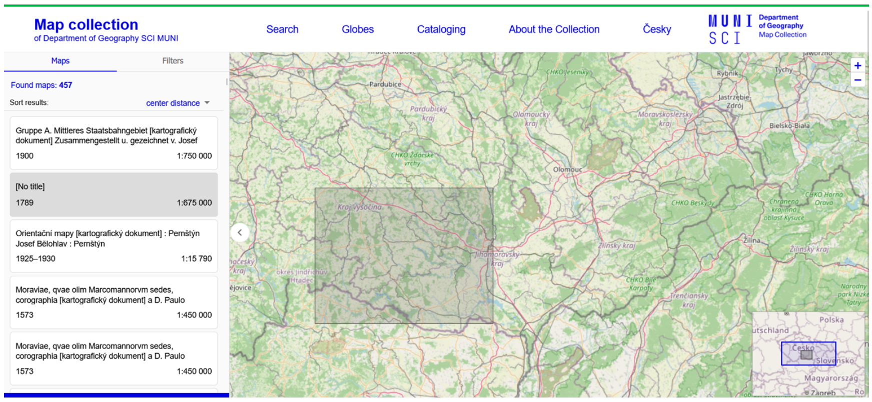


**Figure 6.4:** Screenshot of the homepage with viewport selection mode.

A MiniMap component is rendered as an overlay in the lower-right corner of the main map. It is a separate, OpenLayers instance zoomed to level four. A *moveend* listener on the main map updates the MiniMap's center and zoom on every action. The MiniMap draws the active search area as a coloured rectangle and, when the mouse is hovered over a map record, it draws the area of that map record in a second vector layer.

## 6.7 FilterInfo Component and Extended Filtering

The FilterInfo component is shown in the sidebar on desktop and in the filters tab of the bottom drawer on mobile. It groups all filter controls in one place. In the original application, filter controls were spread across multiple parts of the sidebar with no clear layout.

At the top of the component, the count of all records matching the filters is displayed. Below the count, a toggle switches between rectangle search and viewport search. The active mode is stored in the Redux *searchMode* field.

The coordinate section shows the current search area as four editable text inputs, one for every compass direction. Each input is validated when exiting the text field. Latitude values must be in the range -90 to 90, longitude values in the range -180 to 180, the northern value must be higher than the southern value, and the eastern value must be higher than the western value. When the user edits a coordinate and confirms it, the mode switches to rectangle, and a *rectangleExtentChanged* action is used. This allows the user to type coordinates directly without interacting with the map.

Below the coordinate grid, there are two numeric inputs for the minimum and maximum scale. An error is shown if the lower bound exceeds the upper bound. Furthermore, two text inputs filter records by title and by author strings. All four inputs use Redux actions on each change. The complete filter panel is shown in Figure 6.5.


**Figure 6.5:** Screenshot of the filters tab in the sidebar.

## 6.8 Map Records Overview Extension

The map records list page, shown in Figure 6.6, at `/mr` displays all map records in a paginated table that can be sorted in various ways. The table has five columns: catalogue identifier, title, author, publication year, and scale. Each column header contains a Material UI TableSortLabel component. The first click on a header sorts the rows in ascending order, and a second click sorts them in descending order.

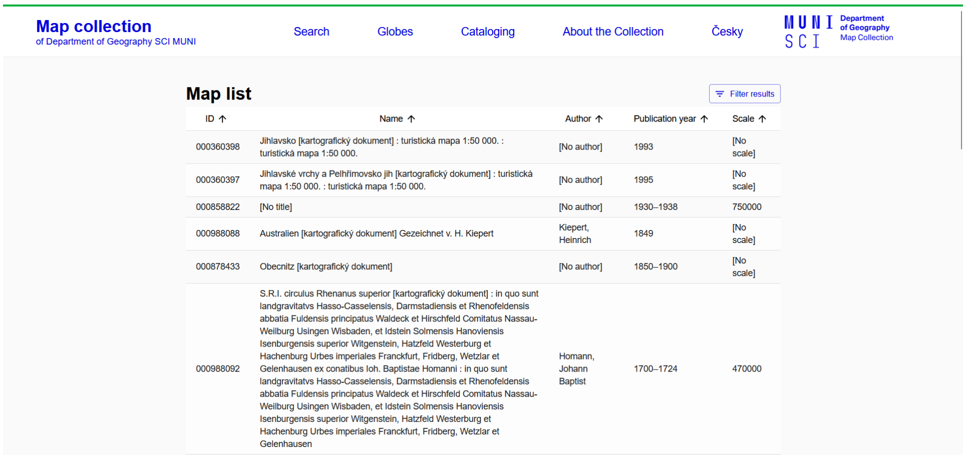


**Figure 6.6:** Screenshot of the redesigned map records page.

Pagination is provided by a TablePagination component. The user can set the page's size to 5, 25, 50, 100, 1000, or 5000 rows. The pagination labels are translatable through the translation system.

A filter dialog was added to the page. A button in the top row opens it. When at least one filter is active, the button appends the count of active filters in parentheses. The dialog contains six fields: title substring, author substring, year from, year to, scale from, and scale to. The fields changes are stored in local component state and do not take effect until the user clicks Apply. This prevents the table from re-filtering on every keystroke. A Clear button resets all fields. The Apply button is disabled when any validations of filtering are not met, such as a lower year bound greater than the upper bound. The complete dialog is shown in Figure 6.7.

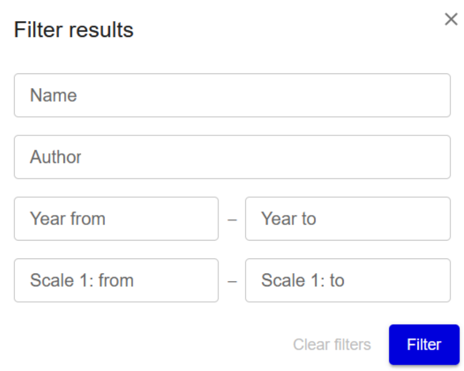


**Figure 6.7:** Screenshot of the map records filter dialog.

The filter logic runs inside a `useMemo` hook and recomputes the filtered rows whenever the applied filters or sort settings change.

## 6.9 Translation System

All user-facing strings in the original application were written directly in Czech inside the TSX files. The upgraded application stores all strings in a single file, `src/core/translations.ts`. The file includes a translations dictionary with two keys, `cs` and `en`, each containing multiple values of strings. The dictionary includes approximately 500 entries.

A helper function, `getTranslation(language, key)`, looks up the string for the given language and key. Every component that renders text calls this function. The active language is stored in a Redux state field. A language switcher in the navigation bar lets the user change it. The switcher uses a *languageChanged* action, which causes all components that include the language selector to re-render with the new strings.

The predecessor application had no English version. Adding bilingual support required translating every string and ensuring that layout elements could accommodate differences in text length between Czech and English. All translations were written manually. Figures 6.8 and 6.9 show an example of the translation. It displays the About the Collection page displayed in English and Czech.

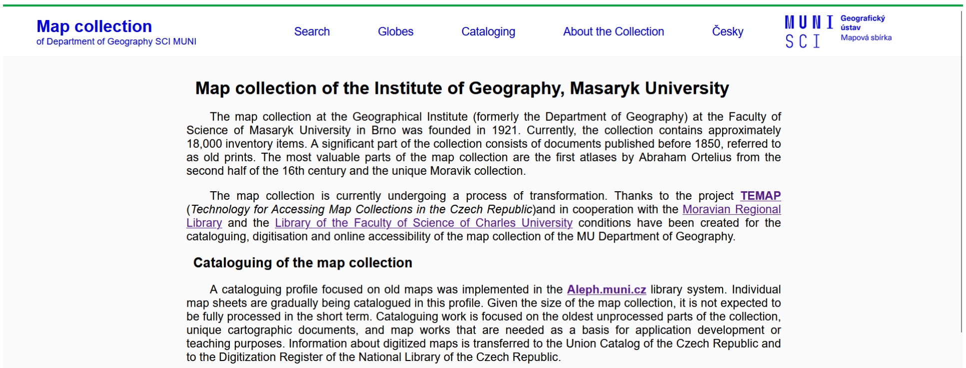


**Figure 6.8:** Screenshot of the *About the Collection* page in English.

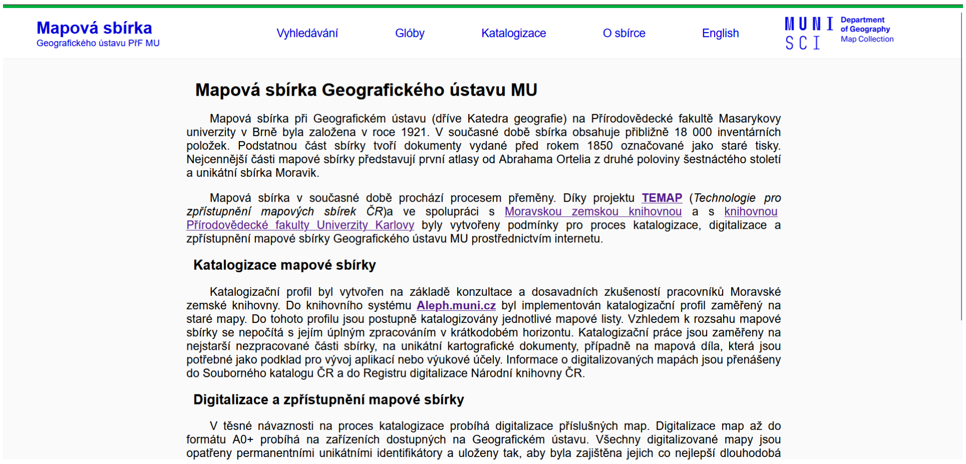


**Figure 6.9:** Screenshot of the *About the Collection* page in Czech.

# 7 User Feedback

Feedback on the upgraded application was collected from three researchers at the Department of Geography, Faculty of Science, Masaryk University. All three respondents are familiar with the original application and use it as part of their work. The feedback was gathered using a structured Google Forms questionnaire with five open-ended questions covering overall impression, visual design, useful features, suggested improvements, and reported errors. The template of this questionnaire is included as attachment to this thesis.

## 7.1 Overall Impression

The overall reception was positive. All three respondents recognized the upgrade as an improvement over the predecessor. One noted that the changes were not immediately obvious and that the new features became apparent only after exploring the application more thoroughly. The other two described the result as objectively better and confirmed that the upgraded version successfully preserves the functionality of the original.

## 7.2 Visual Design

The visual redesign received mixed but mostly positive responses. Two respondents acknowledged that the new design correctly follows the Masaryk University visual identity and is cleaner than the original. One described the original design as more "cozy" despite being visibly outdated and understood the motivation for the change.

One respondent provided more detailed feedback. They noted that the navigation bar layout felt asymmetric and suggested that the page title could be larger. They also proposed adding a historical map image to the header area, adapted to the MUNI color palette, to preserve some of the character of the original. The same respondent suggested adding more blue to prominent interface elements.

## 7.3 Useful Features

The mobile version received the most positive responses. The advanced filters, the viewport search mode toggle, and the placement of the results list in the left sidebar were each mentioned as useful additions by different respondents.

## 7.4 Suggested Improvements

Several suggestions were made for future development. One respondent noted that the search mode toggle is only accessible from the Filters tab and suggested making it available from the Maps tab as well. The same respondent observed that the original application displayed a hint explaining how to move the rectangle and recommended adding an equivalent to the new version.

Another respondent pointed out that the time period filter was more accessible in the original application, where it appeared directly on the main tab without navigating to the Filters section. Since date filtering is the second most common filter criterion after the geographic area, placing it on the main tab was suggested.

Two navigation improvements were also proposed. The Map List button should lead to a filtered view when filters are active, rather than the full unfiltered list. The Found maps counter should function as a clickable link to the corresponding filtered list. One respondent also asked whether bounding box selection is available on mobile, as it was not clear from the interface that the coordinate inputs in the Filters tab provide this capability.

## 7.5 Reported Issues

One respondent found no errors. A second confirmed that the application was fully responsive and correct across all tested screens. A third respondent identified four specific issues.

First, the Czech label for the search shape uses the word for "square" even though the shape could have unequal sides. The correct term is "rectangle". Second, the scale filter does not display the allowed input range as a hint, unlike the time range input, which shows the minimum and maximum ranges for the filter. Third, the time range slider displays overlapping labels when the two handles are set to values close to each other, making the boundary values difficult to read. Fourth, the title and author text filters did not appear to return results in the version tested by this respondent.

# 8 Conclusion

The goal of this thesis was to modernize the frontend of the map collection application at `mapy.geogr.muni.cz`. The work covered the entire frontend layer, from the visual design and component structure to the state management, map interactions, and translation system.

The original application suffered from several problems. The technology stack was outdated, the interface did not work on mobile devices, moving the search rectangle required a keyboard shortcut that touch screens cannot replicate, and the map records page had no text-based filtering or year sorting. The upgraded application addresses all of these issues, and this thesis describes the process of its design and implementation.

Within the implementation, the dependencies Next.js, React, OpenLayers, and Redux Toolkit were upgraded to their current versions. The visual style was changed to follow the Masaryk University corporate design. Footer and a language switcher were added to the navigation bar. The two custom map interactions removed the Ctrl key requirement and made the search rectangle work on touch devices. The sidebar panel was replaced on mobile screens with a swipeable bottom drawer. A centralized translation dictionary made the full interface available in both Czech and English. The *Map records* page gained sortable columns and a filter dialog. The *Search* page gained coordinate inputs, scale and text filters, and a viewport search mode.

The feedback collected in Chapter 7 was mostly positive. Users appreciated the cleaner layout, the mobile support, and the new filtering options. The issues raised were minor and did not concern any core feature.

One area that was not addressed by this thesis, is that many records in the collection belong to multi-sheet map series, and the application still does not show where a given sheet sits within its series, or which other sheets are held. This is outside the scope of this thesis but represents the most useful addition a future version could make.

The upgraded application's source code will be handed over to the Institute of Computer Science of Masaryk University, and the application will be hosted at `mapy.geogr.muni.cz`. It replaces the previous version while keeping the same structure and data sources. The frontend is now responsive, bilingual, and consistent with the university's corporate design.

# Bibliography


[1] Faculty of Science, Charles University Prague. *TEMAP: Technologies for Accessibility of Czech Map Collections*. Faculty of Science, Charles University Prague. 2015. URL: https://web.natur.cuni.cz/gis/temap/ (visited on 05/18/2026).

[2] Klokan Technologies. *OldMapsOnline*. Klokan Technologies GmbH. 2024. URL: https://www.oldmapsonline.org/ (visited on 04/30/2026).

[3] Department of Geography, Faculty of Science, Masaryk University. *Mapová sbírka*. Masaryk University. 2024. URL: https://geogr.sci.muni.cz/o-ustavu/mapova-sbirka (visited on 04/30/2026).

[4] United States Geological Survey. *TopoView*. U.S. Department of the Interior. 2024. URL: https://ngmdb.usgs.gov/topoview/ (visited on 04/29/2026).

[5] National Library of Scotland. *Maps Finder*. National Library of Scotland. 2024. URL: https://maps.nls.uk/geo/find/ (visited on 04/30/2026).

[6] David Rumsey. *David Rumsey Map Collection*. Cartography Associates. 2024. URL: https://www.davidrumsey.com/ (visited on 04/29/2026).

[7] Czech Technical University in Prague. *Czech Historical Atlas*. Czech Technical University in Prague. 2024. URL: https://cha.fsv.cvut.cz/ (visited on 05/02/2026).

[8] MapTiler. *MapTiler Georeferencer*. MapTiler AG. 2024. URL: https://www.maptiler.com/georeferencer/ (visited on 05/02/2026).

[9] Meta Open Source. *React: Quick Start*. Meta Platforms, Inc. 2024. URL: https://react.dev/learn (visited on 05/06/2026).

[10] Vercel. *Next.js: Layouts and Pages*. Vercel Inc. 2024. URL: https://nextjs.org/docs/app/getting-started/layouts-and-pages (visited on 05/06/2026).

[11] Vercel. *Next.js: Lazy Loading*. Vercel Inc. 2024. URL: https://nextjs.org/docs/app/guides/lazy-loading (visited on 05/06/2026).

[12] Microsoft. *The TypeScript Handbook*. Microsoft Corporation. 2024. URL: https://www.typescriptlang.org/docs/handbook/intro.html (visited on 05/07/2026).

[13] Redux Team. *Redux Toolkit: Getting Started*. Redux Team. 2024. URL: https://redux-toolkit.js.org/introduction/getting-started (visited on 05/07/2026).

[14] MUI. *Material UI: Overview*. MUI SAS. 2024. URL: https://mui.com/material-ui/getting-started/ (visited on 05/07/2026).

[15] MUI. *Material UI: Theming*. MUI SAS. 2024. URL: https://mui.com/material-ui/customization/theming/ (visited on 05/07/2026).

[16] Vercel. *Next.js: CSS Modules*. Vercel Inc. 2024. URL: https://nextjs.org/docs/app/getting-started/css (visited on 05/07/2026).

[17] OpenLayers Contributors. *OpenLayers: Quick Start*. OpenLayers Contributors. 2024. URL: https://openlayers.org/doc/quickstart.html (visited on 05/07/2026).

[18] Figma. *Figma Help Center*. Figma, Inc. 2024. URL: https://help.figma.com (visited on 05/07/2026).

[19] Software Freedom Conservancy. *Git: What is Git?* Software Freedom Conservancy. 2024. URL: https://git-scm.com/book/en/v2/Getting-Started-What-is-Git (visited on 05/07/2026).

# A Attachments

In this appendix are listed all attachments submitted with the thesis to the archive of Masaryk University:

- zip file containing the source code of the application.
- folder containing the screenshots of the interfaces of similar works mentioned in the Chapter 2.
- Figma file containing the design of the new version of the application.
- Google Forms form template used for the questionnaire for collecting Feedback, summarized in Chapter 7.