Earth Science Informatics (2026) 19:48
https://doi.org/10.1007/s12145-026-02104-y

RESEARCH

# Tools for packaging static webpages for the development of Multiscale Geo-structural Information Systems (MGS) using Python and open-source software

**Gaetano Ortolano[1] · Gianfranco Di Pietro[2] · Alberto D'Agostino[1] · Roberto Visalli[1] · Eugenio Fazio[1] · Rosaria Ester Musumeci[2] · Rosolino Cirrincione[1]**

Received: 8 November 2025 / Accepted: 12 March 2026 / Published online: 23 March 2026


## Abstract

Geologists produce and analyze various multiscale datasets to reconstruct the kinematics of Earth's deformational processes. The study of tectonic-related rocks (e.g., mylonites) allows for the extraction of geo-structural parameters, which are particularly suitable in this scenario. The use of Multiscale Geo-Structural Information Systems (MGS) is effective for a comprehensive representation of complex geodynamic processes, which can only be achieved through the simultaneous comparison of data from different scales. We propose a methodology for quick packaging of MGS, especially suitable for mylonites, using ad-hoc Python scripts and open-source software. This approach improves connecting information across various scales of observation: from petrographic micro analysis to 3D UAV surveys. We have developed an MGS for studying the mylonites occurring at Palmi Shear Zone (South Italy), one of the most important sites for understanding geodynamics processes of Mediterranean area. MGS have potential applications in territorial planning, resource management, data interoperability and risk analysis.

**Keywords** Earth geodynamic visualization · Multiscalar · Mylonitic rocks · Palmi shear zone · Thin sections · webGIS

Communicated by: Hassan Babaie.

✉ Gianfranco Di Pietro
gianfrancodipietro@gmail.com

Gaetano Ortolano
ortolano@unict.it

Alberto D'Agostino
alberto.dagostino@phd.unict.it

Roberto Visalli
rvisalli@unict.it

Eugenio Fazio
eugenio.fazio@unict.it

Rosaria Ester Musumeci
rosaria.musumeci@unict.it

Rosolino Cirrincione
r.cirrincione@unict.it

[1] Department of Biological Geological and Environmental Sciences, University of Catania, Corso Italia, 57, Catania 95129, CT, Italy

[2] Department of Civil Engineering and Architecture, University of Catania, Cittadella universitaria Via Santa Sofia, 64, Catania 95123, CT, Italy

## Introduction

The current technological landscape is dominated by digital transformation trends such as Cloud Computing, the Internet of Things, and Cyber-Physical Systems. These technologies generate, store, and process massive amounts of data, whose sheer volume and complexity present significant challenges to the effective communication of results of scientific studies. Data visualization bridges this gap by transforming raw data into visual representations that are easier to understand and interpret. This interdisciplinary field leverages visual tools and techniques to effectively communicate information hidden within complex datasets (Lombardo et al. 2018; Walny et al. 2020).

In petro-structural geology studies (i.e., studies based on the extrapolation of quantitative parameters from deformed rocks by tectonic processes), researchers produce a wide range of multiscale data (Ortolano et al. 2020; Fazio et al. 2024) that, if represented through suitable web-based visualizations, can provide a more effective and interactive representation of reality. These include geographic location

and spatial orientation of geological and structural data (i.e., foliation, lineation, joints (Passchier and Trouw 2005) and quantitative petrographic data collected from image analysis of digitized rock thin sections eg. Ortolano et al. (2013, 2014, 2021), Visalli et al. (2021), Roda et al. (2021), Caso et al. (2024). These data can be appropriately characterized as multi-scalar (Melsom 2020), since the scales at which they are generated and produced vary significantly, spanning from the kilometer scale for geographic-related data to the micrometer scale for rock *thin sections* data. Although seemingly unrelated, these data often require concurrent analysis to uncover specific geological insights. This, in turn, underscores the need for a strategy to efficiently represent multiple and multiscale geological data within the same web environment. The variety of geological data translates into different data types, including geographic raster data, non-geographic raster data, and vector data. Raster data consists of arrays indexed by specific coordinates within a coordinate reference system (CRS) for georeferencing (González Canché 2023). While geographical raster data requires precise location referencing, non-geographic raster data, such as optical microscope scans of rock thin sections, faces limitations in georeferencing, making it theoretically and practically infeasible to apply typical geo-cartographic paradigms. This necessitates alternative methods for linking non-geographically referenced data, often via web-based maps or 3D navigators using non-geographic systems. These considerations are equally relevant to vector data, where geometric primitives, defined by vertex coordinates and their associated structures, may be referenced within non-geographic, local Cartesian coordinate systems. In data visualization, especially for web-based geospatial applications, design must prioritize effective communication with the target audience, ensuring at the same time seamless data accessibility. This balance is particularly critical during the post-processing and representation phases, where user-friendly access and clear communication are essential features (Krygier and Wood 2025). When designing tools that convert geological data into visualizations, it is essential to ensure that the resulting data viewers effectively convey information to the end users. Such tools should provide flexible and controllable operators in order to fine-tune the data viewers' output for optimal communication. This is crucial for empowering geology re-searchers who generate data, giving them the autonomy to create web-based visualizations directly without relying on intermediaries (Foerster et al. 2012). It might be difficult to display data and graphics online while referencing several scales and representations. Thematic maps linked to analysis results, 3D, VR, or augmented reality models in combination with spatial statistics are only the simplest use cases of a multiscale approach. In all these cases, it is necessary to contend with different paradigms of graphical representation and the use of different and well-differentiated information technologies for the visualization and interrogation of data by users on the World Wide Web. 3D or Virtual Reality applications, WebGIS, and more complex query systems must coexist in unique web environments. Data scientists who make multiscale datasets often must turn to web providers for visualization on the Web of the results of their analyses (Romero-Organvidez et al. 2024).

This research proposes a methodology facilitating the autonomous publication of multiscale data to a static website, obviating the necessity for reliance on external service providers (Mete and Yomralioglu 2021). By capitalizing on the capabilities afforded by open-source libraries for web development, the potential of crafting tools that facilitate the publication of multi-scalar data on the Web is explored, encompassing both geographic related (i.e. macroscale structural data) and *non-geographic* related (i.e., microscale structural data) information. These tools are not meant to automate the full process of building an entire web environment or web-based application from scratch. They rather provide easy and ready-to-use instruments to quickly convert typical geological and GIS data types, such as raster, vectors and 3D models, into a web-friendly format, using a multi-scalar approach. By implementing a "static-website" generator, a fast and efficient process for Multiscale Geo-Structural (MGS) WebGIS development has been formulated. This ensures the resultant website is rapid and facilitates the seamless publication and dissemination of geodata visualizations. Static websites are a simple and efficient way to present content online. Unlike dynamic websites, which rely on server-side processing, static websites are pre-built and deliver content directly to the user's browser. This makes them a good choice for businesses and individuals who want a fast and easy way to get their information online (Kumari et al. 2023). A schema of the "static webpage" approach in the Web-GIS environment is illustrated in Fig. 1.

## Materials and methods

The challenges faced during the implementation of a "static-website" generator primarily revolved around the need to represent multitype and multiscale geological data simultaneously within the same web view, providing at the same time non-expert users with a friendly way of generating HTML codes. The complexity of handling multi-type and multi-scale geological data is determined by different types of data, structures, and queries.

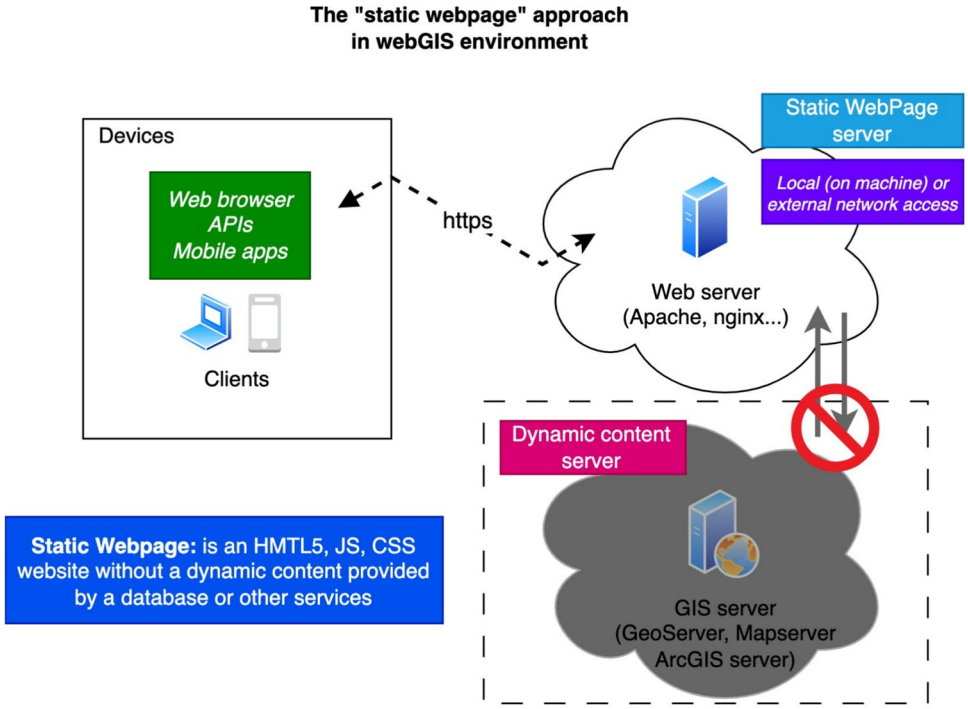


**Fig. 1** Static webpage approach in WebGIS environment

Raster data type contains information referring to a single cell (pixel) defined by specific coordinates and value. It consists of an array that assigns one (or more) values following a precise and continuous 2D type indexing. Defining the location of the array at a precise point in real space is an integral part of the data itself. Indeed, within the geographical domain, it is standard practice to georeference raster data by utilizing pa-rameters contained within a coordinate reference system (CRS) definition file or metadata associated with the specific datum (e.g., GeoTIFF). For geographical raster data, a precise and unique location on a reference datum must be referenced for each cell within the dataset. Several widely used tools also allow coordinate conversion between datums, by considering validity ranges and unavoidable deformations (Snyder 1993). For non-geographic raster data (e.g., optical microscope scans of oriented rocks thin sections), although georeferencing the image into an oriented 3D datum by meticulously acquiring topo-graphic information during sample extraction would be theoretically feasible, the oper-ational complexities and the limited utility of high-accuracy information for geological purposes make such process impractical. It would be possible to assign a precise geo-graphic location like, for example, the exact placement and orientation at the original scale of the rock sample from which a thin section is extracted (Passchier and Trouw 1998, 2005) – i.e., thin sections parallel to the stretching lineation and perpendicular to the mylonitic foliation – XZ section of the finite strain ellipsoid, Fig. 2.

Despite its limitations, particularly in terms of impracticality due to the substantial size difference between minerals seen in thin sections (typically measured in $10^{-6}m$ units) and rock outcrops (which can reach up to $10^{3}m$), this process remains a valuable method for understanding mineralogy. Attempting to define microscale raster data using the same paradigms employed for cartographic raster or sensor/satellite image datasets is ultimately unfeasible from both a theoretical and a practical standpoint. Consequently, there is a necessity to transcend the conventional paradigm that utilizes two-way correspondences between internal coordinates in a raster and coordinates in a geo-cartographic reference system.

This paradigm assumes a bidirectional correspondence between a pixel in the raster with a physically existing point in space; for the reasons stated above, it may not always be applicable or have meaningful results. It must therefore be recognized that there is a threshold of detail beyond which, for the purpose of communicative effectiveness, the notion of a digital twin of a georeferenced object in space is no longer applicable. Consequently, links between information and non-geographically referenced data need to be established. This goal must also be achieved by continuing to use web representations akin to maps or 3D navigators while employing non-geographic reference systems.

The above considerations can also be deemed pertinent in the context of vector datatype. Geometric primitives, defined by the coordinates of vertices and their inherent structures, can be referenced to *non-geographic*, local Cartesian coordinate systems. Consequently, in a multi-scalar model the

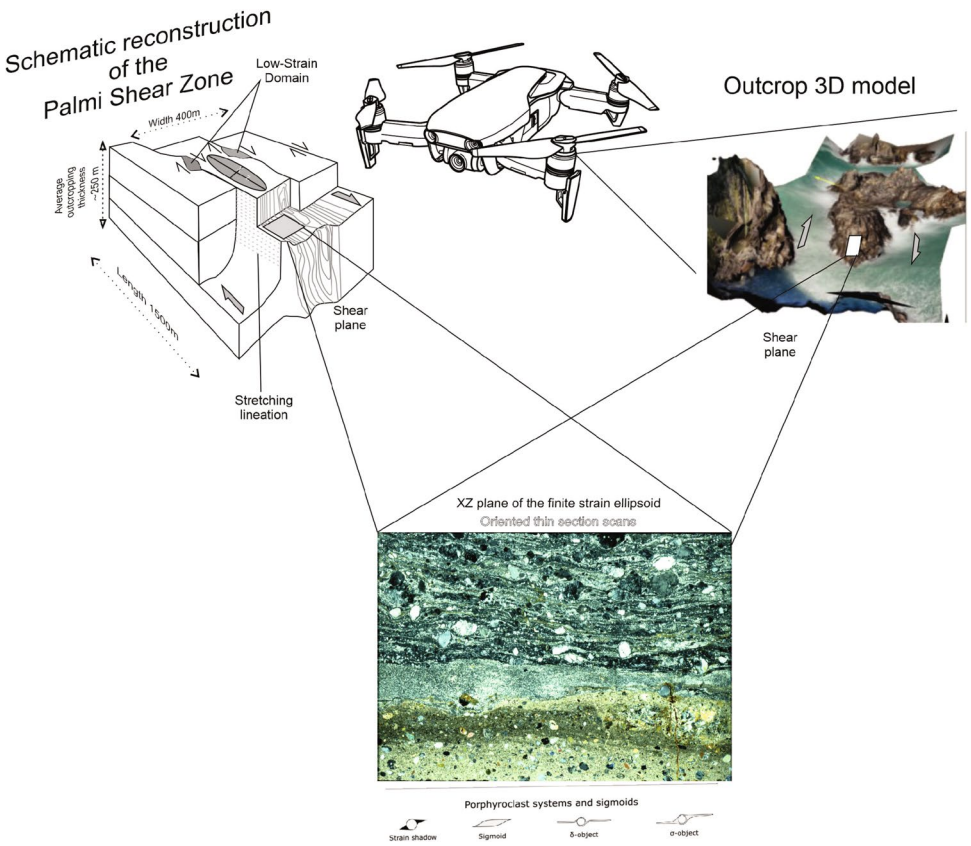


**Fig. 2** [Modified after (Fossen 2016)] - Schematic representation of a multi-scale approach in collecting meso- and micro-structural data from mylonitic outcrops with the aim of unraveling shear sense. For thin sections cut parallel to the stretching lineation, the most common types of shear sense indicators in porphyroclasts are shown

approach of having a single reference system for all data must be bypassed. While the convenience of a unified representation system is sacrificed, there is a distinct communicative advantage in leveraging the diverse array of geometric data representation systems available today. These include 3D models, traditional Geographic Information Systems (GIS), and 2D maps tailored for microscopic or small-scale features. The web development landscape offers numerous open-source li-braries that empower data scientists to readily create web viewers with minimal resources and technical expertise.

Starting with geographic datasets from geological surveys, 3D models from topographic and aerial surveys, and image-sensing analyses conducted on thin sections, it is possible to generate easy-to-publish data visualization and query interfaces with a few steps and basic web development skills. Using some of the most popular JavaScript libraries that are freely available under an open-source license, it is possible to configure a complete framework that contains it.

Thanks to some open-source projects, it is possible to configure a web service for visualizing 3D data and/or 2D vector elements overlaid on images. In specific, the *JavaScript* library *Three.js* (Danchilla 2012) allows the visualization of 3D models within web pages, while the libraries *LeafLet* (Gede 2023; Balla and Gede 2024) and *OpenLayers* allow the visualization and custom-ization of raster and vector data (Wohlmann 2024). Completing this list of tools to design the website are the *Bootstrap* (Cheng 2024) and *JQuery* (Teodorovici 2013) frameworks.

Through *purpose-built* Python algorithms integrable into platforms like QGIS and ArcGIS, pre-configured HTML5 files facilitate rapid web publication for professionals regardless of their technical web development background. This strategy adopts a decentralized, hybrid storage architecture where essential project files and 3D models reside in a local hierarchical directory, prioritizing efficient client-side execution without the need for complex server-side processing. Within this structure, vector feature layers are transformed into GeoJSON format and stored locally to ensure seamless browser integration. To handle high-resolution raster data efficiently, images are subdivided into small, zoom-level-specific tiles following standard conventions. These massive datasets are ideally distributed through low-cost cloud storage solutions such as Amazon AWS-S3, enabling the browser to fetch only the specific data pertinent to the current field of view and thereby optimizing overall system performance.

Numerous possibilities exist for a user not skilled in web development to easily export a map project to a website or WebGIS service, either by taking advantage of paid APIs or open-source JavaScript libraries (Mason 2020). Using all the above considerations, it is possible to propose a unique methodology for the realization of a Multiscale Geo-Structural (MGS) WebGIS. The approach integrates various

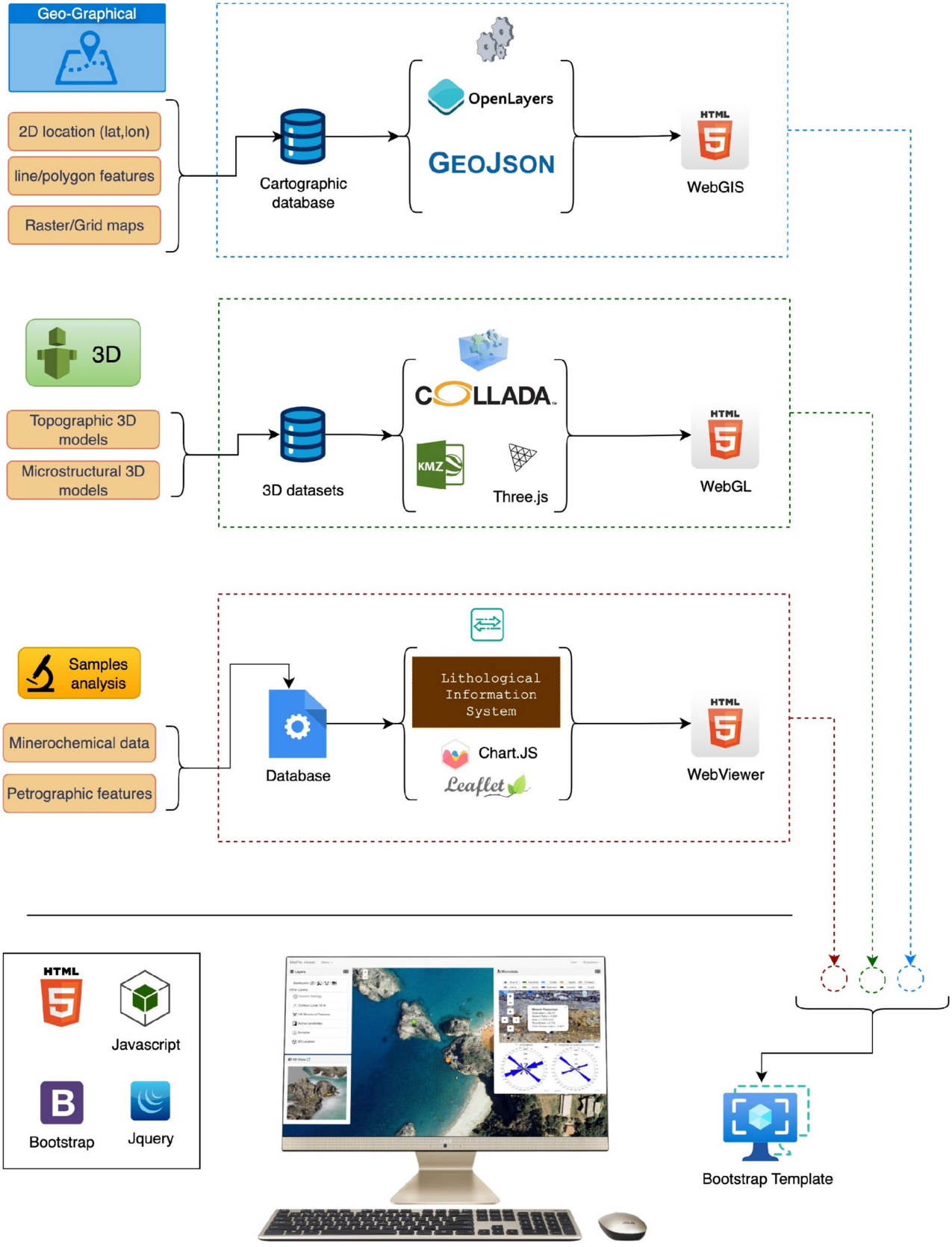


**Fig. 3** Tools and process diagram for packaging static web pages for Multiscale Geo-structural infor-mation System (MGS)

ad-hoc developed tools and open-source software to create static websites that effectively display geographic data in an interactive manner, as displayed in Fig. 3. Three primary tools are utilized for processing and visualization:

1. **Petrographical** data web viewer: using a custom Python toolbox developed in this work (i.e., *LIS_functions.py*), designed to generate an *HTML5* single-page Local Information web viewer developed in this work;
2. **Map** (cartographic data): the popular *qgis2web QGIS plugin* that facilitates the creation of a single-page web viewer directly from geographic data (Duarte et al. 2021);
3. **3D models**: using a custom Python tool developed in this work (i.e., *KMZViewer_functions.py*), leveraging the Three.js open-source library to create 3D KMZ model visualizations within a web viewer (Danchilla 2012).

Each tool outputs HTML single-page code, which is subsequently structured using `<iframe>` tags and additional templates. These templates are further refined through manual customization to meet specific visualization needs. The final static website represents geospatial (and petro-structural) data interactively, using a combination of software and libraries. This workflow demonstrates the seamless integration of advanced tools with web technologies to enhance data accessibility and usability in scientific and educational contexts. This modular approach provides a versatile framework for researchers and developers aiming to represent geospatial data on the web, suitable for an MGS project.

## Petrographical data webviewer

Grain size analyses were performed by processing images of thin sections with image sensing algorithms generating vector data that annotate and describe precise elements within the images (Heilbronner 2000; Tarquini and Armienti 2011; Acevedo Zamora et al. 2024). Grain classification, shape detection, and vectorization of elements from the raster results in vector data that is used to describe or represent the results obtained in the processes (Li et al. 2008; Ortolano et al. 2018, 2021; Visalli et al. 2021).

In a "Microscopic Information System" (Tarquini and Favalli 2010) these data are referred to a non-geographic and non-spatial reference system. The numerical data of the vertices of the vector primitives and pixel indices of the images are certainly representative in terms of scale, but not in terms of position in space. It is possible to know and define the actual size of an element in this system but not its position in space or in a geographic datum. In the case of thin sections, for example, it is important to know the size in millimeters of an image pixel but not its geographical position at the same scale. As illustrated in Fig. 4, the creation of a petrographical data viewer involves a series of steps and features.

A Python library called *LIS_functions.py* has been developed, including seven functions that allow the construction of a "Local Information System" (LIS) of a thin section and the generation of a static website page that is easy to publish and share on the Web. The functions can be used in sequence, as described in Table 1. The steps needed to build a petrographic and data viewer with the developed library are introduced in Fig. 5 and are described in the following subsections.

### Step #1 Prepare datasets and environment

In order to launch the first step of the procedure, a raster file of a high-resolution thin section scan must be available, and a polygon features layer that overlays minerals or other elements of the thin section must have been defined. The Raster data must be a JPG or TIFF file depicting an RGB image of the thin section. The feature layer must be a shapefile, and its polygons must correctly overlay the raster in a no-CRS system. Environment minimum requirements are: *Python* version v.3.11, *GDAL* version v.3.8.5, Python libraries *os, subprocess, bs4/BeautifulSoup* installable via *pip manager*, and *Jupyter*, with *IPython.display* and *ipywidgets* libraries, as computational notebook.

When using shapefiles, the minimum collection of 4 files [*.shp, .dbf, .shx, .cpg*] in ESRI standard format must be provided. The Shapefiles will be converted in GeoJSON format by an *ogr2ogr* subprocess inside the Python environment. The shapefile used as input for data visualization must contain polygons of the minerals occurring thin section, associated with an attribute table populated with the fields described in Table 2. Using the functions `upload_files_widgets()` and `save_to_temp_dir()` inside a Jupyter Notebook, it is possible to launch the process; users can select from the device storage the raster and shapefile to use for LIS generation.

### Step #2: Raster tiles creation

In order to enable more efficient rendering in the user's browser, the display of large raster-type images of thin sections was achieved by generating tile layers. Tile map layers are constructed by subdividing a raster image into numerous small portions at different zoom scales (tiles), with `gdal2tiles.py` (McInerney and Kempeneers 2015), while maintaining the original raster charac-teristics. The individual tiles are saved in a folder structured as `/{zoom-level}/{x}/{y}.png` convention; where the value of "`z`" is a zoom level for web viewer, "`x`" and "`y`" are the coordinates of the tiles points obtained by procedure of subdivision of raster. The browser, depending on the zoom and pan commands provided by the user during navigation, builds

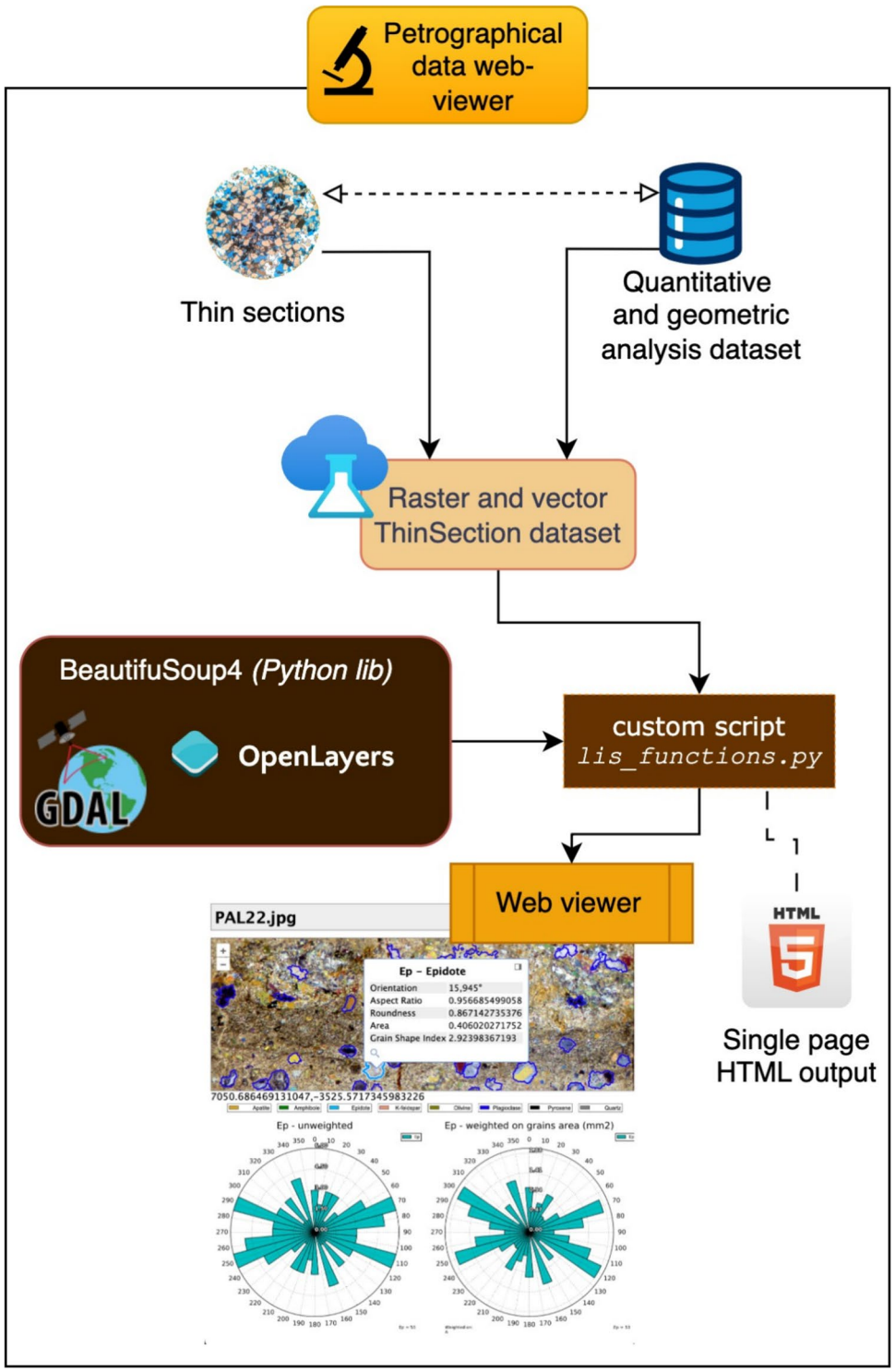


**Fig. 4** Petrographical data web-viewer features and simplified creation process diagram

the strings of tile paths to be used and loads into memory only the needed data. An equivalent of the `gdal2tiles.py` subprocess is also used in QGIS with the native plugin called `tilesxyzdirectory`. Using these sub-processes is important to keep the user as much as possible within the comfort zone of known tools (Baumann et al. 2021).

Given the large number of files generated for these cases, it is convenient to upload them to the Web within free or low-cost file hosting services such as Amazon's AWS-S3 (Han 2015). The function of the `LIS_functions.py` toolbox that operates with `gdal2tiles.py` is called `run_gdal2tiles()`. This function opens a new subprocess in

**Table 1** List of functions used in `LIS_functions.py` library

| Function name | Input/parameters | Output | Description |
|---|---|---|---|
| upload_file_widgets() | N.A. | User Interface with loader for: Shapefile, Jpgfile , projectname | Create a User Interface in a Jupyter Notebook for input files loading |
| save_to_temp_dir() | Shapefile, Jpg file, projectname | Directory path | Save to a temporary directory the files |
| run_gdal2tiles() | Raster path, projectname | Folder with tiles and a basic html map viewer `openlayers.html` | Create tiles from a ras-ter file and a basic html file viewer |
| shp_to_geojson() | Shapefile path, GeoJSON path | GeoJSON file | Convert SHP to GeoJSON by an *ogr2ogr* subprocess |
| add_geojson_overlay() | `openlayers.html` file, project name, geoJSON file | `index.html` file | Add GeoJSON overlay to raster feature using OpenLayers JS and CSS |
| add_popup() | `index.html` and GeoJSON files | `index2.html` file | Add PopUp feature capable to visualize grain data, using *ol-ext* (Viglino 2024) JS and CSS |
| dd_legend_and_rose() | `index2.html` and rose diagram files | `output.html` file | Add legend icons and JS capable to update rose diagrams after user click on grain's polygons |

the Python environment that calls an `ogr2ogr` instance; this last runs the `gdal2tiles.py` procedure using the output of the previous step as input. At the end of elaboration, a folder will be generated, containing the tiles and a simple HTML file (named `"openlayers.html"`) that hold a web viewer of the raster. By default, `ogr2ogr` uses *OpenLayers* Javascript and CSS tools (Steiniger and Hunter 2013). An example of output is illustrated in Fig. 6.

#### Step #3: Adding mineral polygon overlay to the map

In this step, the Python function `add_geojson_overlay_to_gdal2tiles_html_output()` accesses the *openlayers.html* file, already described as the output of Step#2, and changes JS and CSS code to introduce the capability of overlaying the vector polygon layer on the raster tiles in the web viewer. By taking advantage of the Python library *BeautifulSoup*, the *HTML string* is parsed into an object manageable in the Python environment, and then the required new code is added. In particular, the function incorporates JavaScript code that works with *OpenLayers* library for overlaying a *GeoJSON* vector object. Considering that we are operating at a micro-scale and that the raster and vector coordinates lack an established reference system, we assume that all coordinates are assigned in a nominal, fictitious CRS, such as *Pseudo-Mercator* (`EPSG:3857` (Markieta and Rinner 2014), for both a proper map overlay and correct functionality of the WebGIS. This condition is necessary to allow *OpenLayers* (or a similar library) to view and overlay these geospatial data in a webpage, and it only works if the vector layer and the raster layer are already overlaid in a no-CRS representation. This function produces as output another *HTML* file (or an updated version of the previous one) with the specified functionality fully implemented and operational. An example is illustrated in Fig. 7.

#### Step #4: Adding pop-up feature

Within the Python environment used for the proposed methodology, it is possible to access and modify the HTML code obtained in the previous step to add a pop-up feature. A commonly used pop-up functionality in *OpenLayers* is provided by *"OpenLayer Extension"*, also known as "`ol-ext`" (Viglino 2024). The proposed function, called `add_popup_feature()`, modifies the *HTML* file generated in the previous step. Firstly, it adds a link to JS and CSS of `ol-ext` code provided by a Content Delivery Network (CDN) (Vakali and Pallis 2003). The function was implemented using a simple scheme to display the pop-up, by leveraging the `ol.Overlay` class from *OpenLayers*. Then, another function included in the developed JavaScript template, called `getMineralName()`, returns the extended name of the mineral to be inserted in the title bar of the pop-up window, retrieving it from the encoded name in the GeoJSON or source vector data, as described in Table 2.

#### Step #5: Incorporating mineral legends and rose-diagram

In this step, we enhance the web map with an additional layer of customization by incorporating mineral-specific legends and rose diagrams displaying the orientations of the recognized objects (e.g., clasts, grains, voids and pores). Using ArcStereoNet (Ortolano et al. 2021) or other micro-structural analysis software for thin section data, we generated rose diagrams, both *unweighted* and *weighted* on the cumulative area of mineral grains. The diagrams,

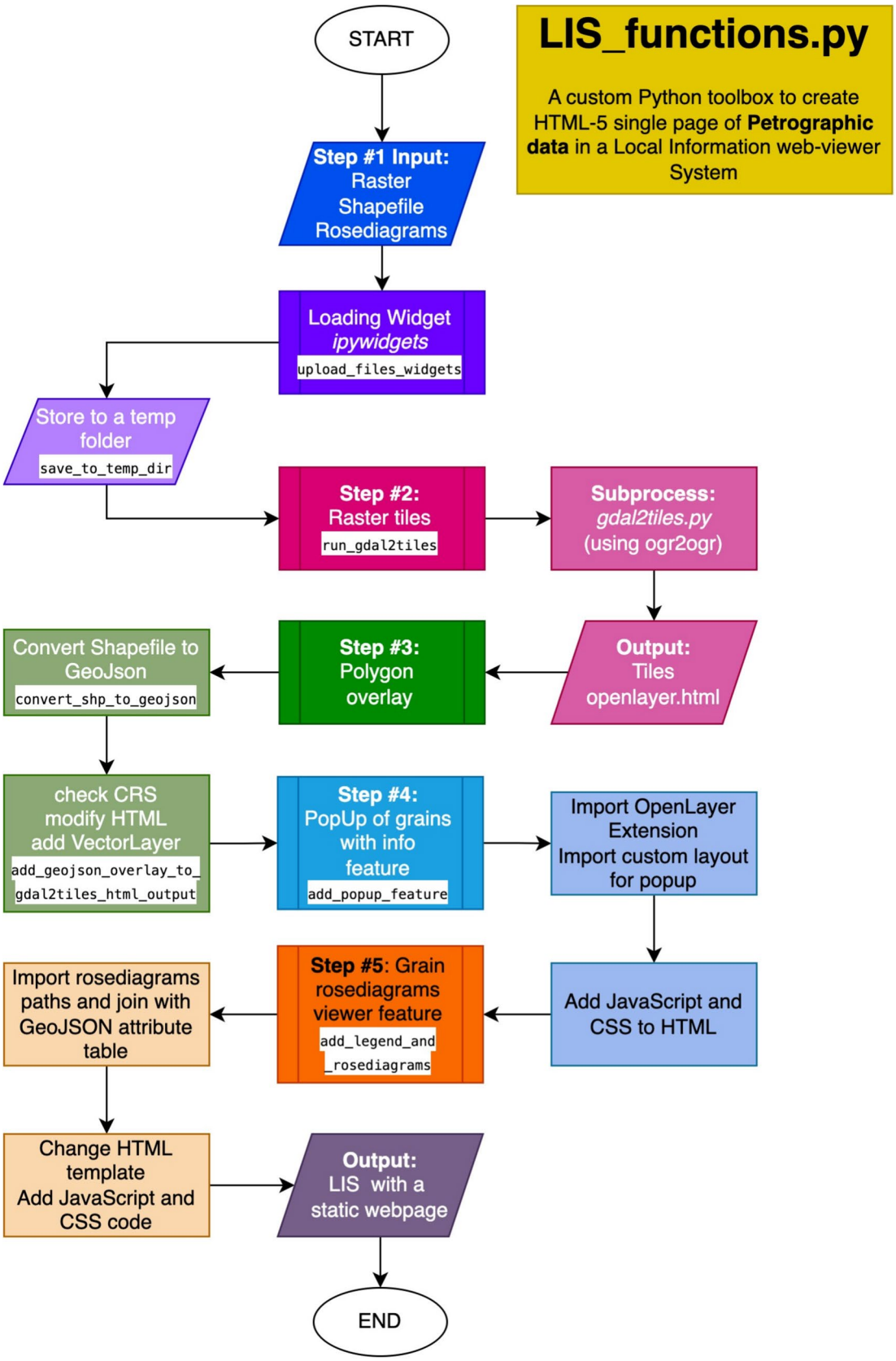


**Fig. 5** Detailed process for production of a petrographic data viewer using `LIS_functions.py` tools

**Table 2** Required fields table of mineral data shapefile used for thin section petrographical data visualization

| Field name | Type | Example | Description |
|---|---|---|---|
| `Mineral` | *string* | `Pl` | Coded name of Mineral |
| `O` | *float* | `90.0` | Degree of Orientation |
| `Asr` | *float* | `0.39843` | Aspect Ratio [%] |
| `A` | *float* | `0.12324` | Area [$\mu m^2$] |
| `R` | *float* | `0.58411` | Roundness Ratio [%] |
| `GSI` | *float* | `1.87833` | Grain Shape Index |

stored in SVG format within the "`/asset/rose`" subfolder, are configured to allow user interaction within the web map interface. Specifically, each user interaction with a polygon classified as a particular mineral (e.g., "*Quartz*") triggers the display of the corresponding rose diagram. To support this feature, an additional functionality allows users to query rose diagram results by clicking on dedicated buttons, each associated to one of the minerals classified within the thin section. Buttons are created with graphic software and saved in "`/asset/legend`" subfolder. The developed function "`add_legend_and_rosediagrams()`" modifies the HTML generated in previous steps as follows:

1. Copies files for rose diagrams and legend buttons into the appropriate subdirectory.
2. Adds a "blank rose diagram" to represent cases where a query returns "other" or no mineral-specific data.

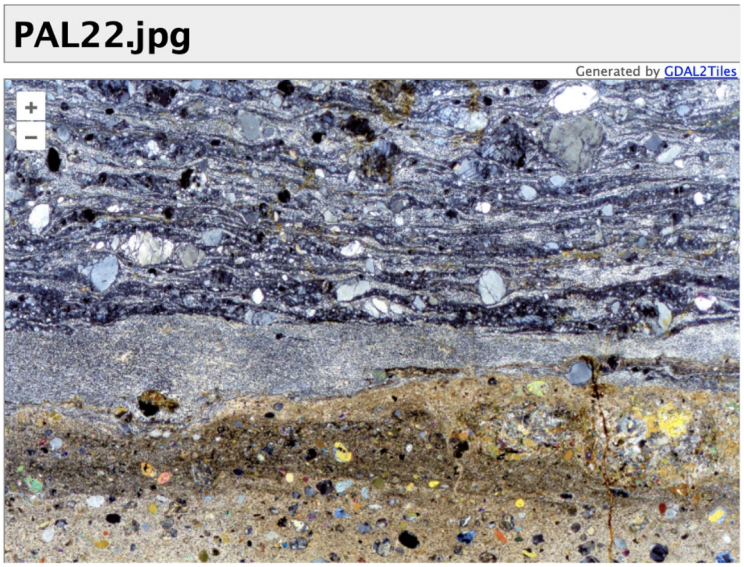


**Fig. 6** The web view of result of Step #2 of petrographic data viewer creation process using `LIS_functions.py` tools

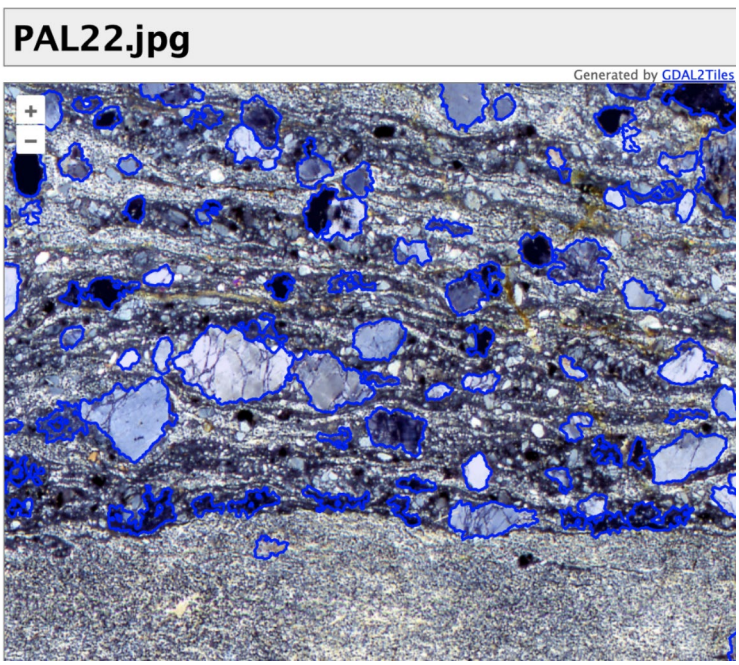


**Fig. 7** The web view result of Step #3 of Petrographic data viewer creation process using `LIS_functions.py` tools, vector grain boundary overlay in the thin section raster data. The grain boundary layer was automatically created with the Micro-Fabric Analyzer tool (Visalli et al. 2021)

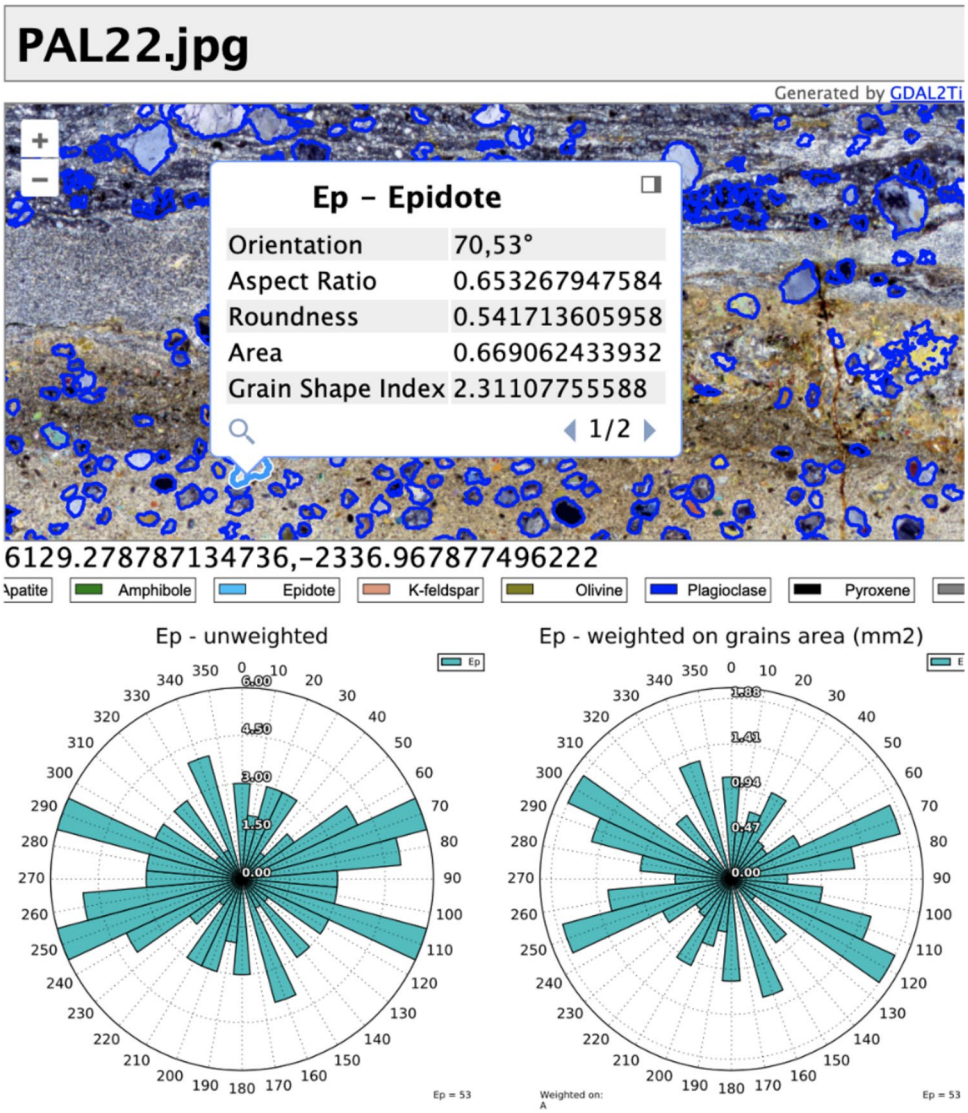


**Fig. 8** Output web view after the final "`add_legend_and_rosediagrams()`" procedure in `LIS_functions.py` library sequence. Users can access specific mineral grain properties of a single grain (polygon appears with a highlighted boundary) and global statistics. A visual representation of mineral grains orientation is provided through rose diagram charts, both *un-weighted* (concentric circles indicates cumulative number of grains) and *weighted* on grains area (concentric circles indicates cumulative area of grains in $mm^2$)

3. Insert necessary JavaScript code and HTML tags for the legends and rose diagrams.
4. Adjusts the map's height and modifies additional style and script settings to optimize display.

The "`add_legend_and_rosediagrams()`" function is designed to handle a list of file paths for both rose diagrams and legend buttons. An example of the result obtained after executing the function is shown in Fig. 8.

## 3D model web viewer

The representation of three-dimensional objects on the Web is done by using 3D rendering libraries of geometric models. The most widely used free and open-source project is *Three.js* (Danchilla 2012), which allows handling raw geometric data and models saved in standard formats. Exporting a three-dimensional geographic model into a website using *Three.js* is a feature already available in QGIS within the *qgis2threejs* plugin, which is used in applications like buildings 3D visualization (Maestri 2017). If data comes from other sources, such as point clouds, models generated by "Structure for Motion" algorithms, or 3D models of microscopic structures, it is possible to take advantage of some modules of the *Three.js* library for displaying files containing the models, also illustrated in Fig. 9.

Inside *Three.js* Library a module called "*KMZ Loader*" allows loading a file of a complex 3D model stored in a Keyhole Markup Language Zipped File (KMZ) file format (Burggraf 2015). KMZ is a ZIP archive file with models, overlays, tiles, texture, images, icons, and other network-linked Keyhole Markup Language (KML) files (Burggraf 2015), all stored in a KMZ file created using the standard wide-used ZIP algorithm. The KML and KMZ can embed

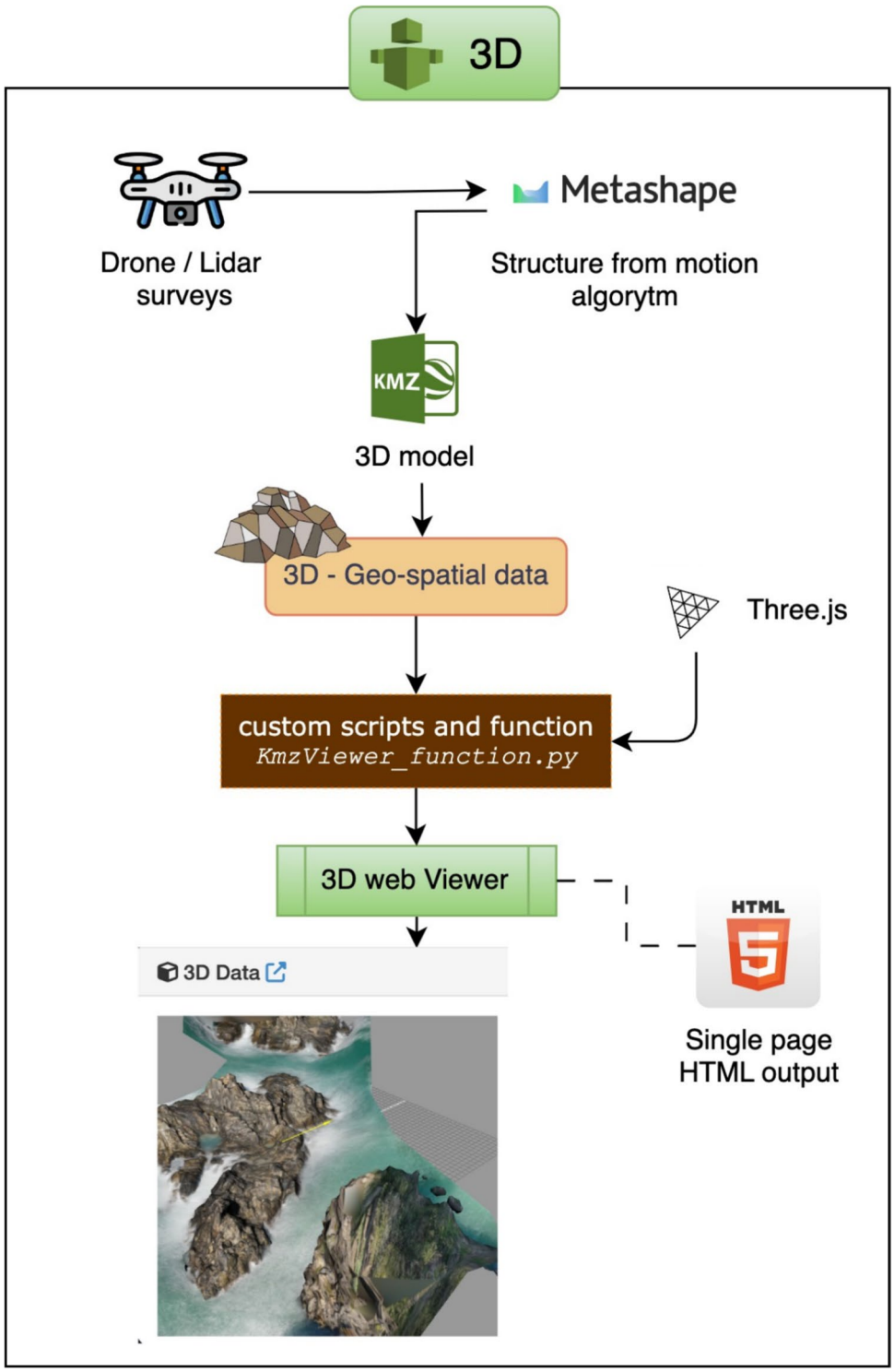


**Fig. 9** The 3D geological models web viewer features and simplified creation process

a COLLADA type 3D model (Arnaud and Barnes 2006), an XML-based schema that facilitates the lossless exchange of 3D assets between various 3D modeling and rendering applications (Preda et al. 2010). A KMZ data package therefore can contain complex 3D models, textures, and additional elements. Export to KMZ or COLLADA format is a widely used feature in most "Structure from Motion" software, such as *Agisoft Metashape* (Rahman and Cahyono 2023). KMZ files extremely reduce the numbers of inputs that must be entered into JavaScript code to display a 3D model.

**Table 3** The sorted list of functions and variables used in the process of creation of a 3D geological model web-viewer using the `KMZViewer_function.py`library

| Item | Variable or function name | Description |
|---|---|---|
| #1 | `KMZ_HTMLHEAD` | Html header, title of page and simple meta tag. |
| #2 | `KMZ_IMPORTLIBS` | Script for import *Three.js* from a CDN server. |
| #3 | `KMZ_THREEJS_ADDONS_FILES_ADD_TO_PROJECT()` | Function to create folder and file for packaging. |
| #4 | `KMZ_OPENINGSCRIPT` | The html row of script that open customized code for viewer. |
| #5 | `KMZ_LOADINGIMAGE()` | Function to create JavaScript code that load a GIF image as a splash screen. |
| #6 | `KMZ_THREEJS_script()`6 | Function to create JS code for loading a KMZ file. |
| #7 | `KMZ_CLOSINGSCRIPT` | String with HTML code for closing script tag. |
| #8 | `KMZ_STYLE` | Html code with nested CSS. |
| #9 | `KMZ_HTMLFOOT` | Html code with closing tag and footer. |

To complete the procedure, the relative position inside the scene must be defined and, optionally, a few default settings for lighting and other rendering features may be tweaked. It is also relatively simple to develop a script that generates an HTML element with a pre-configured viewer for a 3D Model. A flow chart of the proposed methodology related to generating a static web page to display a KMZ model is illustrated in Fig. 9.

If compared with the procedure explained in the previous paragraph, the *HTML* code of a static web page in this case is simpler. By taking advantage of the Python environment, an HTML template including the required JavaScript and CSS libraries was generated. Within the template, variables managed by Python were added, which will be replaced with the processed HTML code. A developed module called "`KMZViewer_function.py`" includes a collection of string variables storing the HTML structure of a 3D model viewer, along with auxiliary functions, as reported in Table 3. Leveraging Python, this structure facilitates the dynamic generation of customized HTML, tailored to specific input data.

To enhance visualization, several essential customizations should be applied to each specific model:

- Define or adjust the custom coordinates for the model's central position, utilizing the JavaScript `KMZLoader` function's parameters.
- Define or adjust the custom coordinates for the camera's center using the parameters of the JavaScript function-`THREE.PerspectiveCamera`.
- Configure appropriate lighting and rendering options.

These modifications are best implemented directly within the *HTML/JavaScript* code, following the generation of the initial HTML structure through Python. Identifying and fine-tuning the specific coordinate values and other variables is a straightforward process within this framework. Using a *Jupyter* computational notebook in a Python environment it is possible to activate all of functions and variables in Table 3 using the `create_3dml_viewer()` method, which in turn takes as input three elements: Name of the project, KMZ file, Splash-screeen image. An example of simplified Graphical User Interface (GUI) for input file loading inside a Jupyter Notebook is illustrated in Fig. 10.

The output folder, named after the project name contains: an HTML file with the "viewer" app, the loaded KMZ file, the splash-screen image provided, a subfolder called "`/three`" with *Three.js* library files embedded. Due to the inherent limitations of Content Delivery Networks (CDNs) in efficiently serving these specific files, and to mitigate potential issues arising from Cross-Origin Resource Sharing (CORS) restrictions, a preferred strategy is to store and retrieve the required files locally within the KMZ model file (Hossain 2014). The "`/three`" subfolder contain addons and other JavaScript files imported from the *Three.js* library. These files are downloaded from the *Three.js* repository and are organized to maintain the default structure of the *Three.js* framework. Specifically, they are stored within the directory "`three/addons/jsm`", which contains the following subfolders:

- `three/addons/jsm/controls`: Contains the OrbitControls module, enabling navigation features for 3D scenes.
- `three/addons/jsm/libs`: Includes the *fflate* module, a fast JavaScript compression/decompression library designed to enhance rendering speed.
- `three/addons/jsm/loaders`: Contains the *ColladaLoader*, *KMZLoader*, and *TGALoader* modules for loading respective file types within the *Three.js* environment.

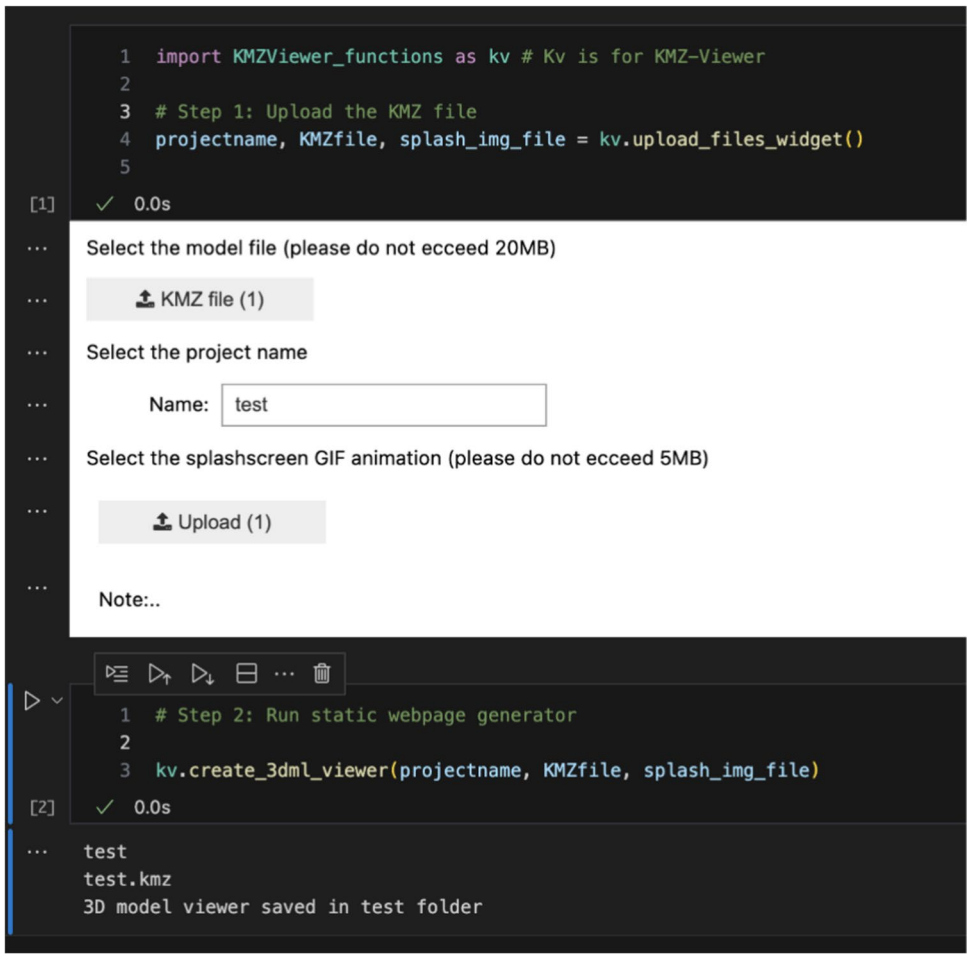


**Fig. 10** Jupyter notebook interface with the only two steps required to create a web-viewer static page of a KMZ file, using the developed `KMZViewer_function.py` tool

An example of output web-viewer in a `<iframe>` HTML element is illustrated in Fig. 11.

## Geographical and maps data viewer

With the same approach, geographic data visualizers and online maps can be generated by taking advantage of the capabilities offered by the *OpenLayers* library. A quick view of creation process is illustrated in Fig. 12. QGIS community offers several plugins and tools to generate web maps (Duarte et al. 2021); for instance, *qgis2web* can export an entire map project, with all graphic textures to a folder with files for publication on the web. This procedure is straightforward and requires at most a few corrective operations for displaying the data. This method of publishing is used by geospatial scientists for various types of maps (Azmi et al. 2022; Bachri et al. 2022; Maguire and Longley 2005) and is a common way to obtain a static web page with several features. A schematic view of the described process is represented in Fig. 12.

## Merging multiple data webviewers in an MGS website

The realization of an integrated MGS data visualizer, as revealed by the analysis conducted, can be achieved through the synergistic use of the Bootstrap framework and `<iframe>` tags (Cheng 2024). This approach, geared toward modularity and efficiency, involves the definition of a web layout optimized in terms of User Experience (UX) and User Interface (UI). In this context, UX plays a key role in ensuring viewer usability and accessibility. An effective UX results in an intuitive and satisfying user experience (Visser et al. 2018), characterized by:

- *Clarity*: information must be presented clearly and concisely, facilitating understanding of displayed data;
- *Efficiency*: interaction with the viewer should be smooth and intuitive, allowing the user to achieve their goals quickly and effectively;

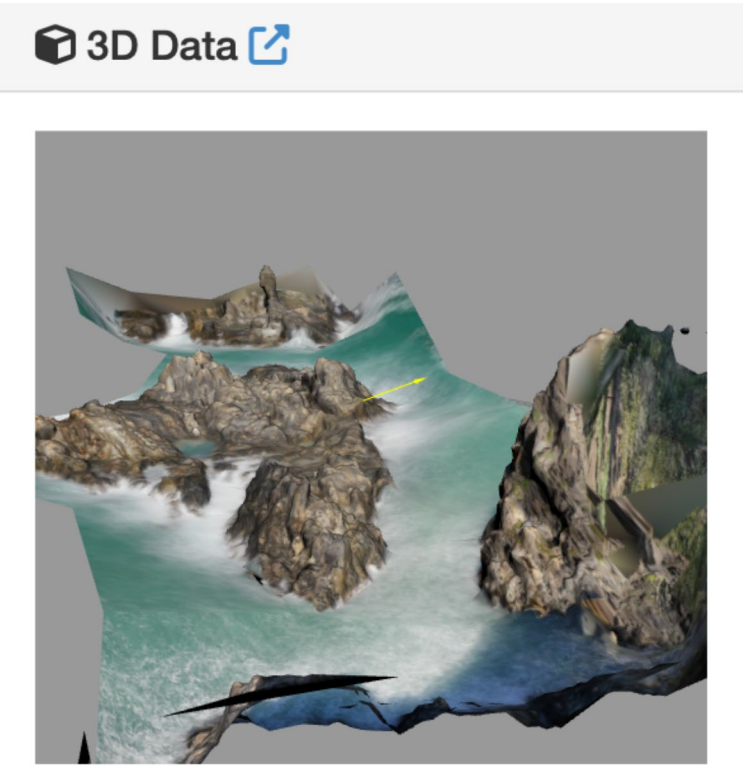


**Fig. 11** Output web view of a 3D model using `KMZ-viewer_functions.py`. The yellow arrow represents North direction

- *Consistency*: the design and functionality of the viewer must be consistent across sections and with established web interaction conventions;
- *Accessibility*: the viewer must be accessible to all users, including those with disabilities, through the implementation of specific functionality and the adoption of web accessibility standards.

The UI, in turn, contributes significantly to the communicative effectiveness of the viewer. A well-designed UI (Norman 2013) is distinguished by:

- *Aesthetics*: the visual appearance of the viewer should be neat and appealing, stimulating user interest and promoting data exploration;
- *Organization*: graphical elements and controls must be arranged in a logical and intuitive manner, guiding the user in interacting with the viewer;
- *Feedback*: the viewer must provide clear and timely feedback to user actions, con-firming successful interaction and preventing errors;
- *Responsiveness*: the design of the viewer must adapt dynamically to different screen sizes and resolutions, ensuring an optimal user experience on any device.

The design of an integrated *geoscience* data visualizer requires careful consideration of UX and UI aspects. Effective synergy between these two elements results in a positive user experience that promotes understanding and analysis of geoscientific data. The HTML standards and scripting languages allow different ways of merging parts from different sources (Simon 2023). The simplest, but not the best performing way of combining a web page with its attachments within a frame of another web page is the use of the `<iframe>` tag. In this proposed methodology, which has as one of its goals to simplify the engineering process, embedding methods that could offer more effective performance by taking advantage of code refactor and optimization of all embedded scripts and libraries (Li and Zhang 2021) have been left out.

It is possible to combine in a web template the three datatypes frames: *Bootstrap* (Naik and Naik 2024; Elrom 2016) streamlines UI development with its ready-to-use components and adaptable grid system, ensuring an optimized visualization on any device. *Bootstrap* grids are a powerful layout system provided by the *Bootstrap* framework, which is a popular front-end toolkit for building responsive and mobile-first websites and web applications. The grid system in *Bootstrap* is based on a 12-column layout, which allows developers to create flexible and responsive designs that adapt to different screen sizes and devices. *jQuery* simplifies client-side scripting with its intuitive functions for *DOM* manipulation and other common tasks, providing a friendly way of adding interactive elements to a website or web application. Interactive elements like sliders, accordions, modal dialogs, and animations can improve user experience by engaging users and making the interface more dynamic (Naik and Naik 2024).

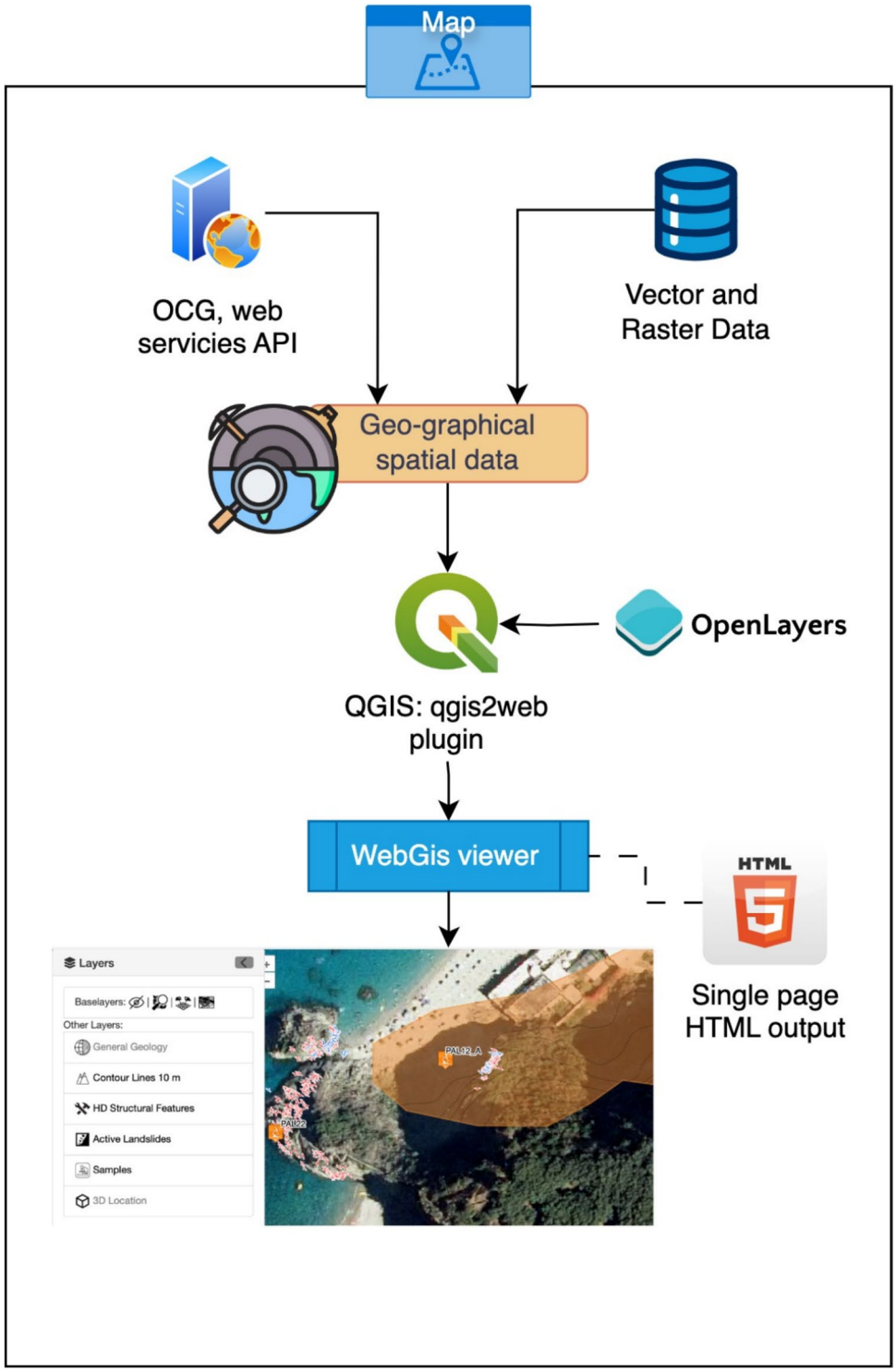


**Fig. 12** Geo-graphical and maps data viewer features and creation process schema

In this work, as further described in the Section "Case study", we organized the components into a hierarchical structure of files and directories. In this process, the use of *Bootstrap* was crucial in building the layout of the viewer, particularly through its system of "containers". Containers in *Bootstrap* are fundamental elements that allow the content within a web page to be organized and structured (Elrom 2016). They provide a predefined and responsive area that can automatically adapt to different screen sizes, within which layout elements, such as text, images, tables, and, in our case, multiscale *geodata* viewers, can be arranged.

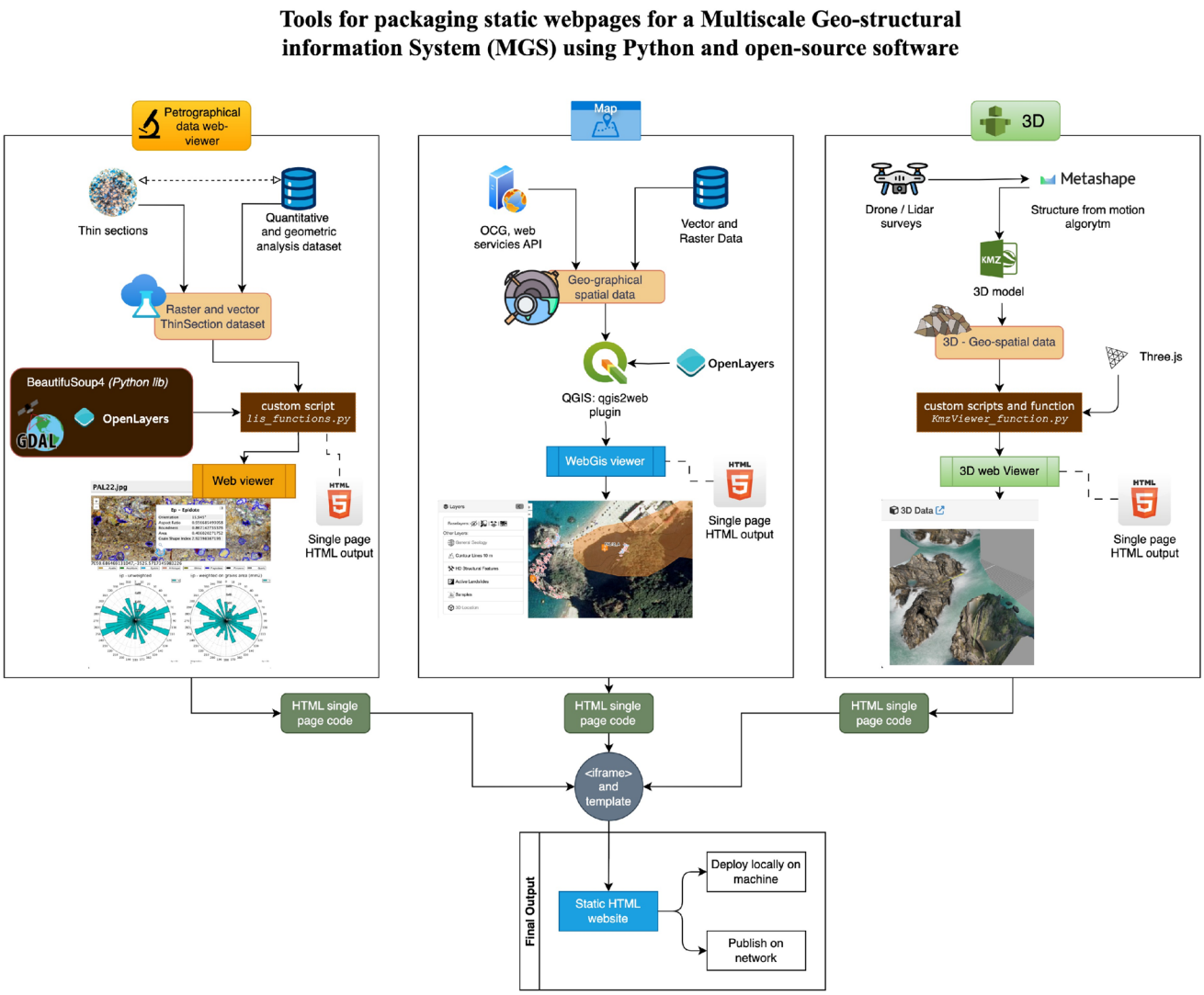


**Fig. 13** Global diagram for packaging a Multiscale Geo-structural information System (MGS) web viewer features and making-of process schema

Within containers, layout elements are organized using *Bootstrap*'s grid system, based on rows *(.row)* and columns *(.col)*. The columns, in turn, can be nested to create complex and articulated layouts. In our specific case, `<iframe>` tags, which include the individual viewers, are placed within these containers and organized via the grid system. This approach provides high flexibility in the arrangement of the viewers and a responsive layout that adapts to different screen sizes. The integration of the independently developed viewers within a structured layout with *Bootstrap* allows for a combination of development flexibility and design modularity. In addition, the use of `<iframe>` tags for data visualization ensures effective functional isolation between components, allowing individual visualizers to be updated without impacting the overall website structure. An example of an integration of three static webpage data viewer for an MGS website process is represented in Fig. 13.

This methodology, if properly validated through cross-browser and cross-device compatibility testing, profiles as an efficient and scalable solution for creating modular web platforms for geoscientific data visualization and analysis.

## Case study

The *Palmi Shear Zone* (PSZ) is located on the Tyrrhenian side of the southern offshoot of the Calabrian coast (Fig. 14a). From a lithological point of view, the backbone of the study area is characterized by crystalline basement rocks, mostly formed during the Hercynian orogenetic processes in a period ranging from the upper Paleozoic up to the Permi-an-Triassic transition between about 325 and 290 Ma (Cirrincione et al. 2015; Ortolano et al. 2020; Fazio et al. 2024), and partially

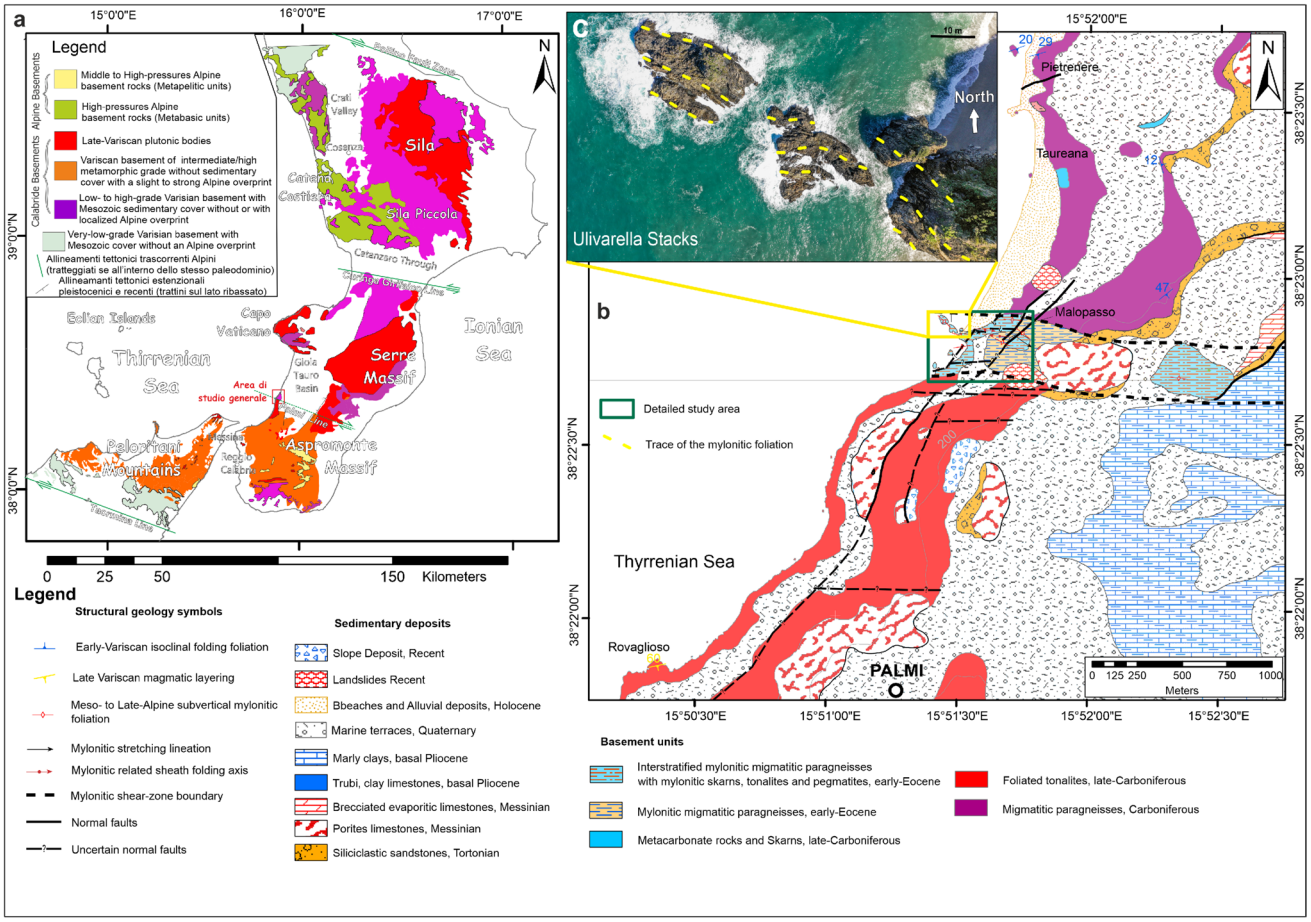


**Fig. 14** [Modified after (Fazio et al. 2024)] – Geological location of the mylonitic rocks outcrop chosen as test site for the Web-GIS development. **a**) Geological sketch map of the basement rocks forming the Calabrian Peloritani Orogen; **b**) Geological map of the general study area and location of the detailed study area; **c**) panoramic view of the Ulivarella stacks with trace of the subvertical mylonitic foliation

covered in basal discordance by a sedimentary sequence aging from the Tortonian up to Holocene (Fig. 14b).

Basement rocks consist mostly of tonalites, represented by the Monte Sant'Elia, and migmatitic paragneiss, outcropping in the inner part of the Taureana waterfront, at the foot of the archaeological site of the "*Taureani*" (Fig. 14b). Subordinately, it is possible also to observe metacarbonate layers, locally metasomatized, pegmatitic dykes that cross-cut tonalites, and migmatitic paragneisses. From a geological-structural point of view, most of the basement rocks are charac-terized by a strong tectonic-related fabric. The migmatitic paragneiss is charac-terized by a main schistosity connected to an isoclinal folding stage; the tonalites are characterized by a magmatic layering, probably linked to the tectonic control of the late-Hercynian emplacement mechanisms (Passchier and Trouw 2005; Russo et al. 2023).

In correspondence with the *Ulivarella* stacks, recently framed within the *geosite Italian national inventory*, the Hercynian-age-related fabric is abruptly interrupted by a sub-vertical mylonitic-related foliation, where an intense plastic-type deformational activity is observed (Fig. 13b). The sub-vertical foliation is the result of a tectonic process known in the geological literature as "*mylonitization*", a geological process where the rate of strain and the rate of recovery counterbalance each other without one taking over from the other (Wise et al. 1984).

These conditions often form good kinematic indicators known as porphyroclasts, whose study can provide precious information about the mechanisms that gave rise to the deformation itself (Xypolias 2010), allowing the reconstruction of past geological kinematics and contributing significantly to the geodynamic reconstruction of the past tectonic realms (Fazio et al. 2024; Ortolano et al. 2020).

Generally, the mylonites of the PSZ show evidence of a pervasive plastic deformation that was generated from about 60 and up to about 40 Ma during the Paleocene-Eocene transition (Prosser et al. 2003) along predominantly strike-slip tectonics (Fazio et al. 2024; Ortolano et al. 2013,

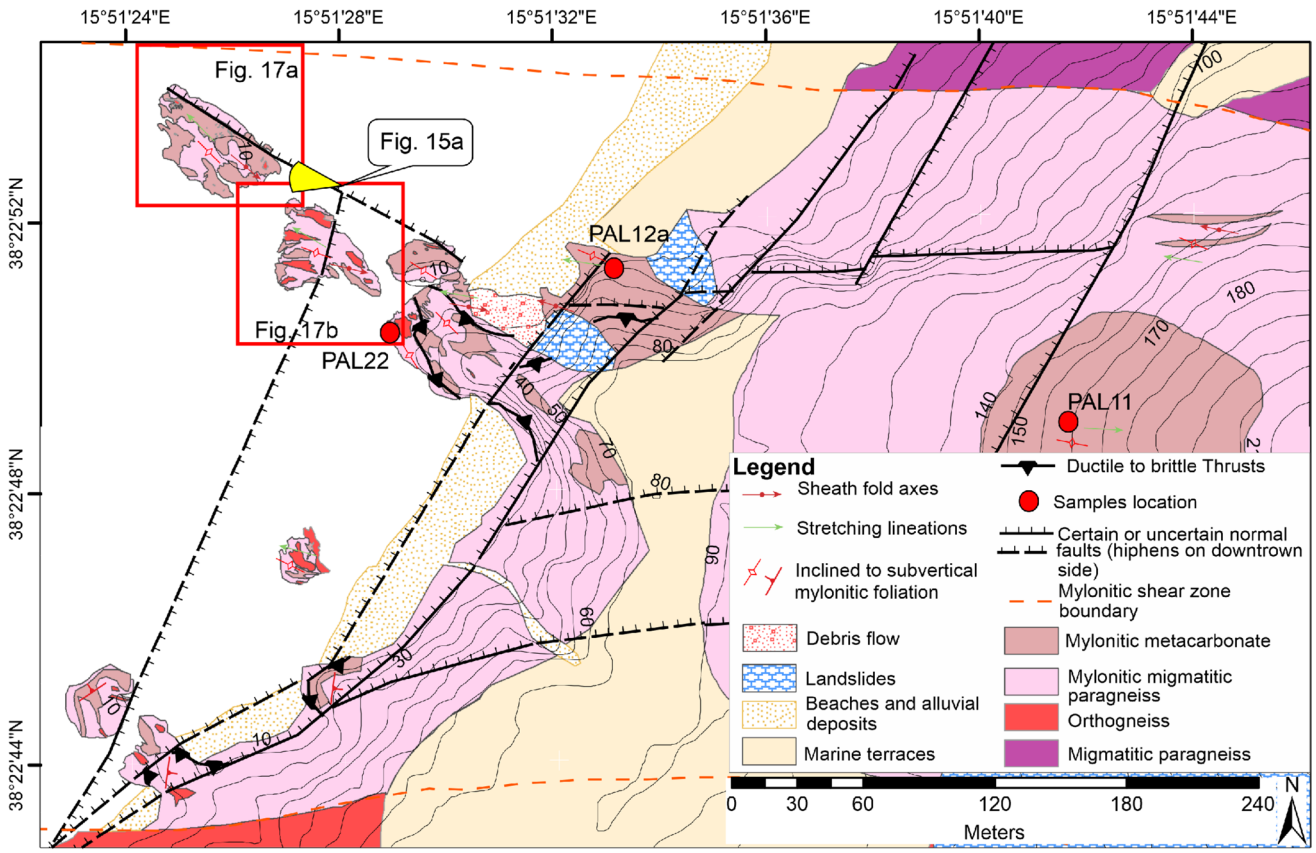


**Fig. 15** [After (Fazio et al. 2024)] - Structural geological map of the detailed study area chosen as test site for the Web-GIS development

2020; Prosser et al. 2003). As already mentioned, at the Ulivarella stacks and along the portion of the beach in front of the stacks themselves (Figs. 14b, c, 15), it is possible to observe that tonalites and migmatitic paragneisses, originally characterized by sub-horizontal related fabrics, are abruptly interrupted by a sub-vertical mylonitic-related fabric, rheologically controlled by the ductile deformation of metacarbonate layers.

These meta-carbonate layers have driven most of the plastic deformation due to the mylonitic process, acting as a highly plastic medium within which pieces of tonalites and paragneiss maintained a relatively rigid deformational behavior, forming thousands of porphyroclasts of various sizes (Fig. 16).

According to Cirrincione et al. (2015), this shear zone represents the deepest piece of a more complex strike-slip tectonic alignment known as the Palmi Line, which has driven, along with other sub-vertical crustal-scale strike-slip tectonic structures such as the *Sardinia Balearic Shear Zone* (SBSZ), the separation of the Calabrian microplate from the Sardinian-Corsica block, contributing to the rising of the so-called *Calabrian Peloritani Orogen* (CPO) (Fig. 17).

The geological-structural study of mylonites implies the necessity to observe all the tectonic-related structures developed at all scales of observations, from the outcrop scale up to the micro-scale and sometimes also to the nano-scale (Mamtani 2025). The mylonites of the PSZ employed in this work as case study were previously studied both at the *micro-scale*, through the sequential combined use of two semi-automated GIS-based procedures (Ortolano et al. 2014, 2021) able to extract quantitative structural parameters at the thin section scale (Ortolano et al. 2020), and at the *outcrop scale*, through UAV surveys (Fazio et al. 2024), using in sequence the software *Cloudcompare* to extrapolate the orientation of geological structure from 3D model (Thiele et al. 2017) and the software GeoVis3D (Vasuki et al. 2014; Tavani et al. 2024) for the extrapolation of the structural attitudes from the 3D model (Fazio et al. 2024). Finally, structural-related features at the outcrop and micro-scale have been projected, respectively, as lower hemisphere *stereographic projection* and as rose diagrams displaying porphiroclasts orientations, using the software ArcStereoNet (Ortolano et al. 2021). The steeply dipping cliff along the Palmi coastline as well as the articulated structural features observable along the

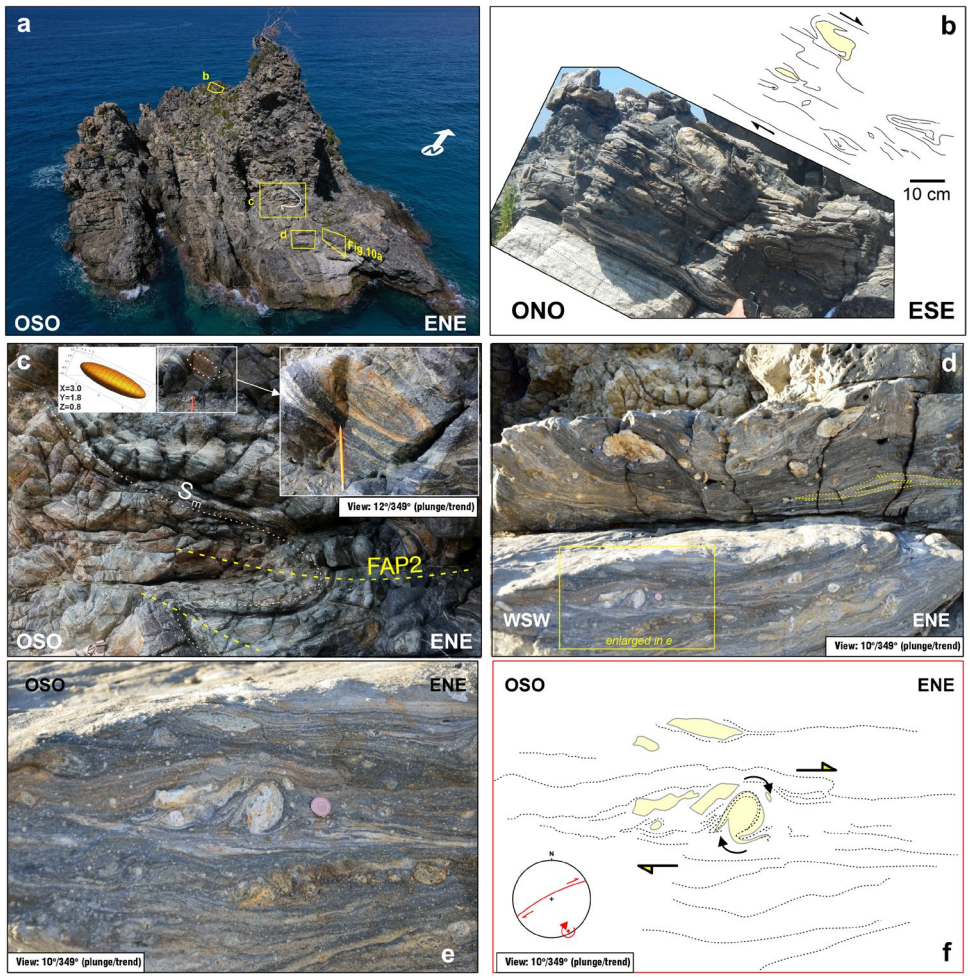


**Fig. 16** [After (Fazio et al. 2024)] **a**) Panoramic view of the most far Ulivarella Stack from the beach; **b**) Ultramylonitic layer developed at the expenses of the migmatitic paragneisses and of the metacarbonate rocks (upper dark layer); **c**) sheath fold in the upper limb of a metric fold (FAP2); **d**) vertical surface of outcrop with a dextrally-sheared sheath fold and several clasts displaying opposite shear sense; **e**) clast indicating top-to-the-ENE shear sense; **f**) interpretation of structures depicted in (**e**); the inset shows a stereoplot with location of a clockwise vorticity axis compatible with the clast at the center of the picture

PSZ outcrops required, indeed, an aerial survey to assist in the field geological mapping and to acquire more robust orientation geological datasets (Fazio et al. 2024). For the first-time publication of this Multiscale Geological-Structural Information System (MGS) in a WebGIS platform, it was decided to allow the interactive visualization of the 3D model of the *Ulivarella* stacks, characterized by well-preserved structural features related to the deep-seated mylonitic activity of the PSZ (Fig. 18).

Three thin section samples, cut parallel to the stretching lineation, here W-E oriented, and perpendicular to the mylonitic foliation, have been selected to interactively visualize the microstructural-derived features of the mylonitic process. Specifically, *PAL11* is representative of the mylonitic evolution on the migmatitic paragneiss, *PAL12a* is useful to depict the rheological behavior of the mylonitic metacarbonate rocks, and finally *PAL22* is representative of the mylonitic orthogneiss (Fig. 19).

The automatic segmentation of the mineral grain constituents via optical high resolution thin section scans, together with the classification of the mineral phases via X-ray map analyses, allowed obtaining three distinct classified images (Fig. 20). These images were used, in turn, to isolate the porphyroclasts, subdivided by mineral type, from the rest of the rock constituents (Fig. 20).

By taking advantage of the method of the minimum bounding box (Visalli et al. 2021), this last step allowed, in turn, for the generation of both weighted (i.e., normalized to the area of mineral grains) and unweighted rose diagrams (Ortolano et al. 2021), which display the orientation of the long axis of the classified porphyroclasts. The obtained plots were then exported as SVG images to be used into the MGS Web-GIS platform (Fig. 8).

In Fig. 20 is represented the final view of MGS web-viewer developed using the methodology illustrated in this work. The code and data are available on GitHub in the *MGSTools* public repository https://github.com/gianfrancodp/MGStools and a running webpage is pub-lished online https://gianfrancodp.github.io/metpet/ .

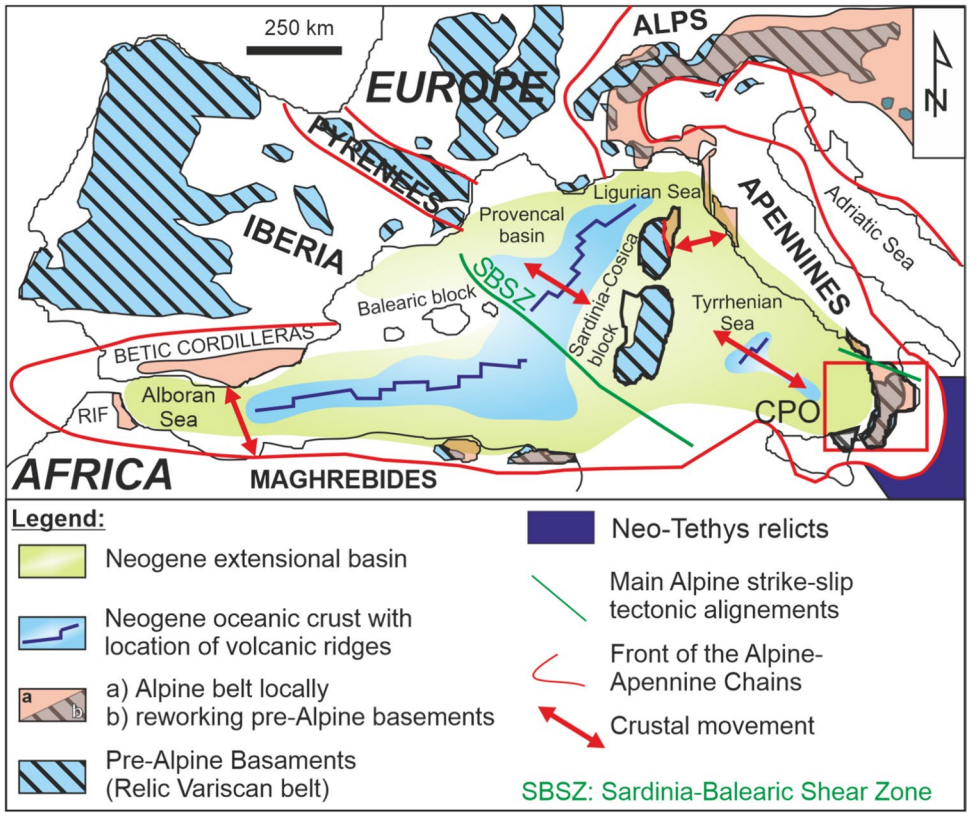


**Fig. 17** [Modified after (Cirrincione et al. 2015)] - Present-day distribution of the Alpine and Pre-Alpine Basement in western Mediterranean realm with CPO location and main Alpine strike-slip tectonic alignment

## Discussion

Digital transition of spatial geodata is one of the most challenging aims in the framework of the international innovative strategies (Autant-Bernard et al. 2013); data interoperability and intelligibility through geoportals development are among of the primary tools of such objective achievements (Maguire and Longley 2005). This is even more true for modern geosciences (Baru and Lin 2009; Lombardo et al. 2018), which in the last years have been increasingly integrating digitized geological data from different sources from macro- (e.g., Geophysical surveys; LIDAR and UAV data acquisition; Structural investigations) to micro-scales (e.g., Optical and Electron microscopy; Raman spectroscopy; Microtomography). With the main aim of developing a simplified methodology for the digital transition of geological datasets at different scales of observation and from different sources, in this work we wanted to focus the attention on the integration of multiscale geo-structural data on an interoperable web platform taking, as case study one of the most informative and well preserved mylonitic strike-slip shear zone for the western Mediterranean Geodynamic reconstruction (i.e., the Palmi Shear Zone).

PSZ operated since the early-Eocene, as a crustal-scale binary facilitating the lateral juxtaposition of some Calabrian-Peloritani micro terranes, contributing to drive at first the strike-slip dislocation of the Sardianian-Corsica block, from the Balearic one, and after to facilitate the opening of the Thyrrenian Sea basin (Cirrincione et al. 2015; Ortolano et al. 2020; Fazio et al. 2024).

Starting from a geological basemap that serves as the main entry point to the WebGIS interface, geo-referenced rock samples and 3D outcrop models can be accessed and explored. Rock samples are connected to their own microstructural datasets, populated by data resulting from quantitative petrographical analysis at the thin section analyses, where each mineral grain is opportunely vectorized, classified and characterized by its own fabric-related features (Visalli et al. 2021). Digital outcrop models, derived for example from UAV surveys, are explorable through dedicated 3D views, thereby allowing their interactive navigation (Fazio et al. 2024; Tavani et al. 2024).

The integration of multiscale geodata characterized by diverse datatypes, is at the base of a Multiscale Geostructural Information System (MGS), which improves connecting information across various scales of observation. An MGS aims at enabling integrated, multiscale data

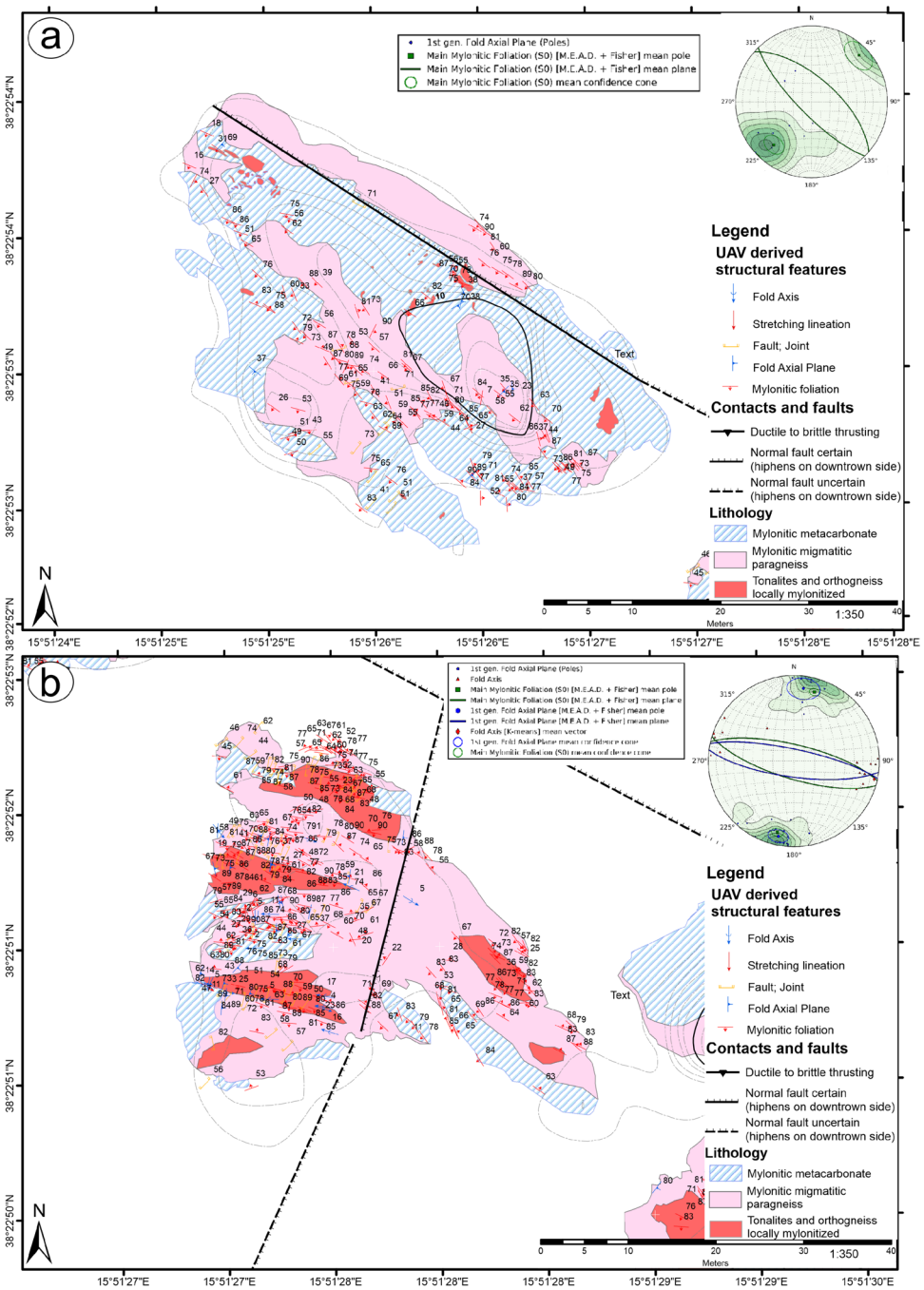


**Fig. 18** [After (Fazio et al. 2024)] structural maps of the **a**) Ulivarella Stack 1, and **b**) Ulivarella Stack 2, derived from UAV surveys

exploration, which, in turn, may reveal kinematic and petro-genetic relationships otherwise hidden by isolated datasets. The connection on a single static web platform of microscale features directly to macro-scale ones can be more effective than analyzing individual layers or non-connected data. This multiscale layer connection is also not automatically achievable through dedicated GIS software solutions, due to the intrinsic scale differences of geological data and incompatibilities with common reference systems. In this view, the developed tool and the proposed methodological workflow allow packaging multiscale geological data into HTML-based file formats compatible for web publishing through semi-automated routines. In order to opportunely employ these tools, properly organized and standardized data need to be provided.

Micro-structural data packaging tools (e.g., `LIS_functions.py`) are specialized for processing data derived from rock thin section analyses, including several

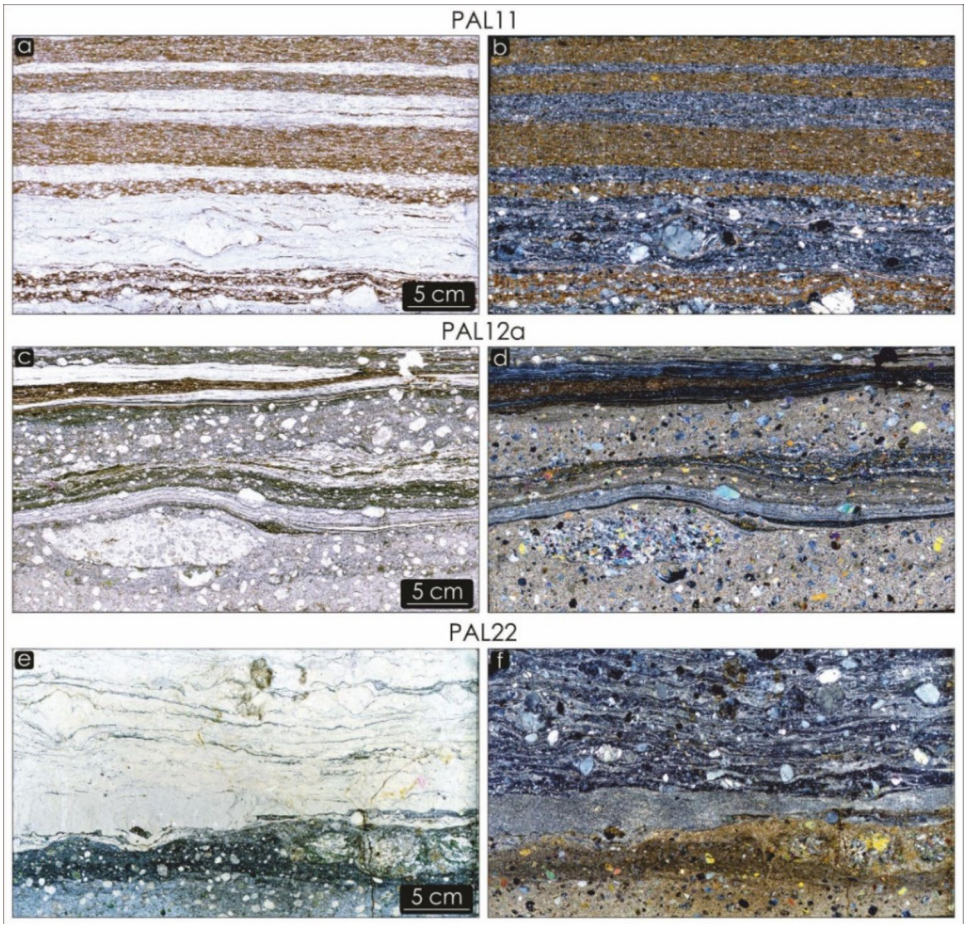


**Fig. 19** High resolution thin section scans: **a**,**c**,**e**) Plane-polarized Light; **b**.**d**.**f**) Crossed-polarized Light

functions dedicated to the construction of a "Local Information System". In order to launch the procedure, a raster file (JPG or TIFF) of a high-resolution thin section scan must be available, and a polygon vector layer (Esri shapefile) of digitized minerals or other elements of the thin section must have been defined, including in its attribute table the fields described in Table 2. The main challenges that this suite of tools addresses is achieving a proper overlay of raster and vector data on a no-CRS system and tiling raster data for a more efficient rendering in the user's browser. Optionally, the procedure allows integrating a legend that display mineral names, and on-click events that can trigger the visualization of pop-ups and dedicated graphics, which must be provided as SVG image files (e.g., rose diagrams displaying the orientation of selected mineral grains).

Packaging of three-dimensional objects is simplified in the library `KMZviewer_functions.py`, that requires 3D models to be provided as KMZ, a widely compatible file format. Using a Jupyter computational notebook in a Python environment it is possible to quickly run the routine, as illustrated in Fig. 10. To complete the procedure, the relative position inside the scene must be defined and, optionally, a few default settings for lighting and other rendering features can be tweaked.

The custom Python library `LIS_functions.py` was employed for the creation of static web pages populated with micro-structural data (Ortolano et al. 2020). Such data, consisting of mineral classifications and microfabric statistics, derived from analyses conducted on thin sections collected from three geo-referenced rock samples. The combined use of software dedicated to the automated classification and vectorization of rock thin sections – i.e., Q-XRMA (Ortolano et al. 2018) and MFA (Visalli et al. 2021) -– permitted to generate opportunely digitized rock thin sections datasets, characterized by overlayed raster and vector layers. In our specific case, the thin sections employed for the first MGS website were collected from mylonites and are opportunely oriented and cut parallel to the stretching lineation and orthogonal with respect to the main mylonitic foliation (Passchier and Trouw 1998, 2005). Mylonitic rocks are of particular importance within the frame of structural geology study due to their specific petrogenetic process, characterized by a relatively high rotational deformation with very

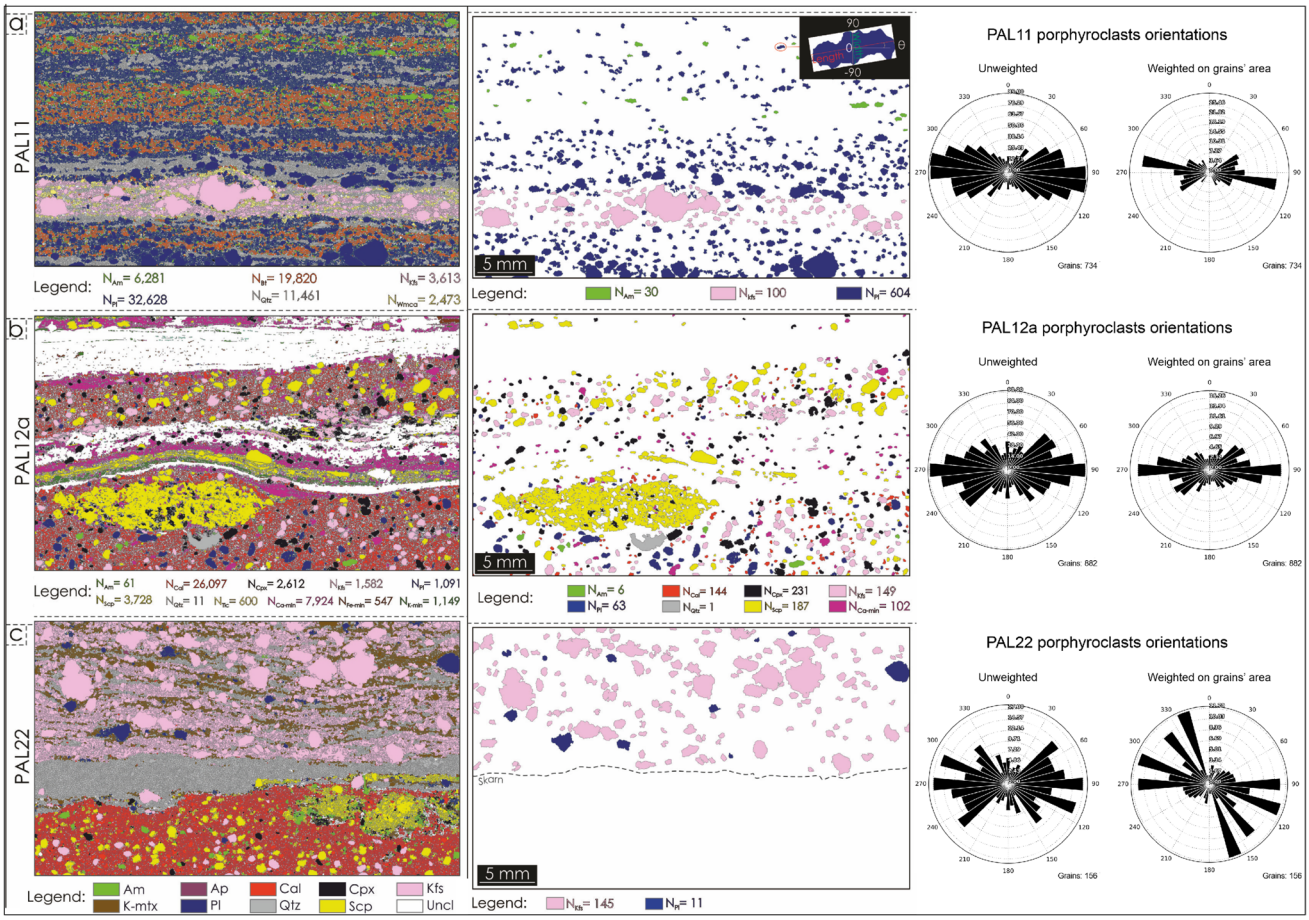


**Fig. 20** [*Modified after* (Ortolano et al. 2020)]. Mineral phase classification with mineral grain size distribution outputs and distribution of the porphyroclast domains with their relative rose diagrams for: **a**) *PAL11*; **b**) *PAL12a*; and **c**) *PAL22*

scarce recrystallisation and in a constant steady state deformational process without loss of primary cohesion. This implies that these types of tectonites (Wise et al. 1984) are suitable to extrapolate very useful fabric-related parameters (Xypolias 2010) and then to unravel the kinematics of the deformational events, such as those reconstructed through the analysis of porphyroclasts orientations using the software ArcStereoNet (Ortolano et al. 2021).

Next, using the textttKMZviewer_functions.py library, a 3D outcrop model obtained via UAV survey campaign by Fazio et al. (2024), was included in the static web page. Then with *qgis2web* plugin a basic WebGIS map was created, including geological features and elements collected in various surveys on the case history location. In the final step, using the Bootstrap framework and `<iframe>` HTML tag, a unique website that contain an MGS for the Palmi Shear Zone was published (https://gianfrancodp.github.io/metpet/) , as illustrated in Fig. 21.

The methodology employed has therefore enabled us to achieve a simplified and completely open-source process capable of building geoportals without the aid of server management tools, within which, it is possible to interactively navigate georeferenced data on geological maps, 3D models of significant outcrops, as well as high-resolution thin section scans, accompanied by vector elements that quantitatively characterize the fabric of the analyzed tectonites. This result is important not only for sharing geological, structural, and microstructural information among experts, but also for developing a greater understanding of the rocks analyzed (i.e., tectonics) beyond the confines of pure Earth scientists, thus opening new horizons toward greater interoperability and intelligibility of multiscale geological data.

## Limitation and future works

While the current implementation serves as a robust Proof of Concept (PoC), large-scale distribution requires specific data management and performance considerations.

The management of software dependencies and version compatibility ensures long-term functionality through environment standardization and local resource embedding. The

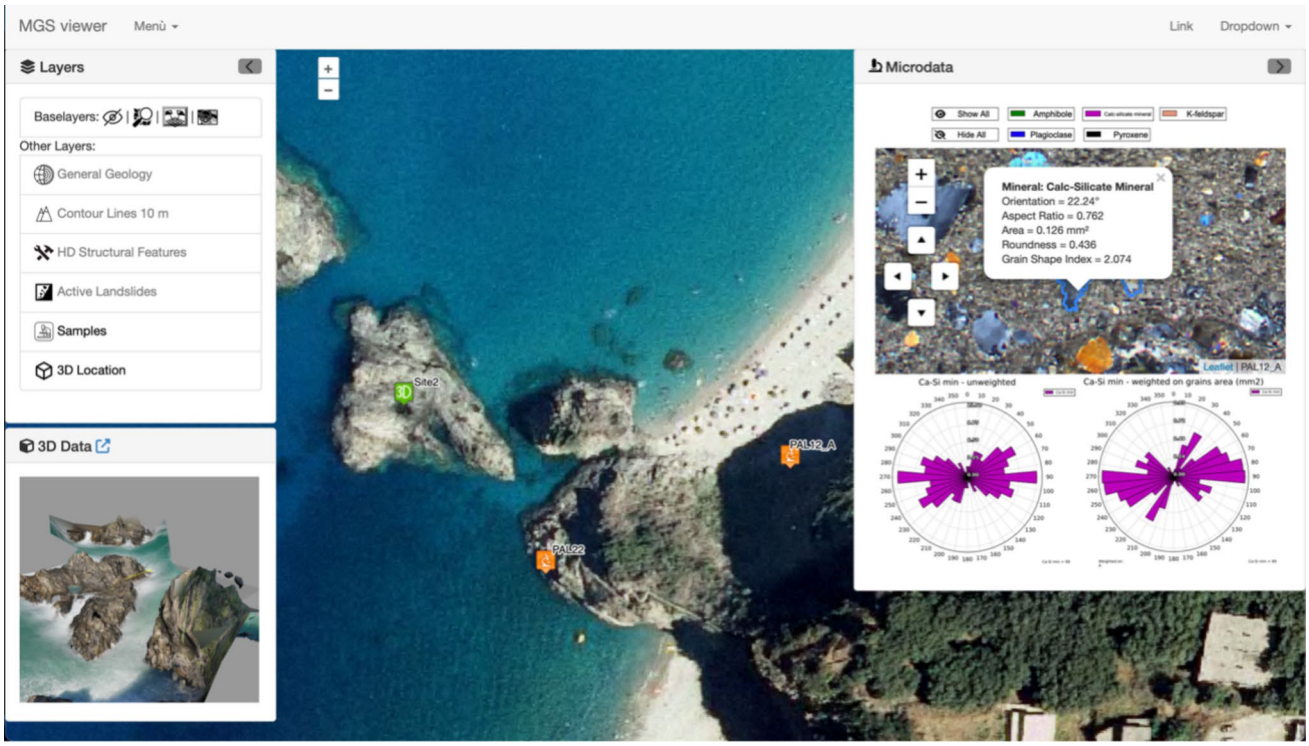


**Fig. 21** Screenshot of the MGS viewer of the mylonitic Palmi Shear Zone taken as an example for simultaneous, multiscale visualization of structural geological data useful for scientific, professional, and geo-tourism purposes

core system requires Python 3.11 and GDAL 3.8.5, with auxiliary libraries like BeautifulSoup and ipywidgets managed via the pip manager. To facilitate robust web-based visualization using frameworks such as Three.js, OpenLayers, and Bootstrap, the methodology recommends storing essential assets locally rather than relying solely on Content Delivery Networks (CDNs). This strategy, combined with a static webpage architecture and the use of Jupyter Notebooks for documented data processing, mitigates risks from server-side shifts and cross-origin (CORS) restrictions. While these measures reinforce reproducibility, future maintenance may still require refactoring the custom `LIS_functions.py` and `KMZViewer_functions.py` libraries if browser standards or the underlying Python and GDAL ecosystems undergo significant changes.

The developed MGSTools in this work creates static pages, so performance largely depends on the user's hardware r00esources (browser and device). This *client-side* limitation can be addressed by transitioning toward a distributed architecture optimized for large-scale geospatial data broadcasting.

To effectively handle high-resolution thin section scans, the system uses the `gdal2tiles.py`, a *raster-tiling* algorithm to subdivide images into a structured folder of small "tiles." This ensures that the browser only loads the data necessary for the current zoom level and view.

In this work, a developed as a PoC, an initial approach in this direction was hypothesized by utilizing Amazon AWS-S3 servers for the distribution of the orthophoto layer. By implementing a raster tile strategy—where high-resolution imagery is subdivided into a pyramid of small, zoom-level-specific tiles—the system ensures that the browser only loads the data necessary for the current field of view. This significantly mitigates the processing burden on the user's device. For large-scale distribution of the MGS system, however, further code refactoring and the implementation of a dedicated back-end system designed specifically for broadcasting geospatial data are required. Such an evolution would allow for more efficient data streaming and management, ensuring that the system remains responsive even as the volume and complexity of the geological datasets increase.

Finally, using `<iframe>` tags within a Bootstrap framework allows different data viewers (3D, 2D, and microscopic) to be updated or refactored independently without disrupting the site's overall structure.

The addressing the future transition from a proof of concept (PoC) to a system optimized for widespread distribution, a lot of development will be needed: a code refactory in order to consolidate and ebbed the scripts to improve loading speeds on clients, a back-end implementation with a dedicated system designed for broadcasting geospatial data and a major upgrade in geo-data interoperability.

The potential for utilization extends beyond academic research into several critical domains:

- **Geotechnical Engineering and Infrastructure:** Providing on-site engineers with immediate, multiscale access to structural data, from regional fault maps to grain-scale micro-fracture analysis.
- **Public Outreach and Education:** Offering an intuitive, browser-based interface that democratizes access to complex geological information without requiring specialized GIS software.
- **Institutional Risk Management:** Allowing local authorities to host centralized, high-performance datasets that can be accessed by diverse stakeholders during planning phases.

To realize these possibilities, future development will prioritize code refactoring to improve loading speeds, the implementation of a dedicated back-end for geospatial broadcasting, and a significant upgrade in data interoperability standards.

In this context, the MGS system emerges as more than a visualization tool; it represents a crucial component for the integration of advanced decision-support instruments. Such a Multiscale Geo-Structural Information System can provide comprehensive and real-time effective communication of hazards, regulations, and the technical reasoning that leads to critical urban decisions within a Smart City Development approach (Miles et al. 2024; Wei et al. 2024). By bridging the gap between micro-scale geological evidence and macro-scale urban planning, the system fosters a transparent environment where scientific data directly informs public safety and sustainable development.

## Conclusion

The quantitative reconstruction of relative movements between and within tectonic plates or micro-plates is increasingly correlated with careful multiscale geological-structural analysis, capable of integrating geo-structural data can span from the kilometer scale to the micrometer scale. Although seemingly unrelated, these data often require concurrent analysis to uncover specific geological insights, underscoring the need for a strategy to efficiently represent multiscale geological data within the same environment. In this view, a challenge that must be addressed is to build an interlinked but dynamic environment that allows a seamless navigation from micro-scale (e.g., microfabric of rock thin sections) up to macro-scale structural data (e.g., 3D digital outcrop models). This, in turn, requires solving technical issues related to the interoperability of data characterized by different reference systems, file formats and scales of observation.

The developed tools presented in this work provide a standardized and simplified solution to achieve a seamless integration of multiscale data within the same web platform. While not directly providing a fully automatic routine for developing a WebGIS from scratch, they allow packaging geodata into file formats compatible for web publishing, laying the foundation for the creation of a Multiscale Geo-structural Information System (MGS) through the generation of static web pages. After online publishing, the resulting MGS WebGIS can be easily accessed by end users through a common web browser. Documentation and a user guide were also prepared and published in the MGS GitHub repository https://github.com/gianfrancodp/MGStools. The implementation of the Palmi Shear Zone MGS, provided as case study, demonstrates that by leveraging open-source libraries for web development, the proposed methodology allowed a seamless integration of multiscale geo-structural data related to the mylonitic rocks characterizing the study area, particularly important within the frame of structural geology study due to their specific petrogenetic process, suitable to unravel the kinematics of deformational events. A running live webview of the MGS build for the case history is available at the website: https://gianfrancodp.github.io/metpet.

In conclusion, the web-based MGS presented in this study represents a significant advancement in the realm of digital twins for territorial analysis, owing to its capability to manage and visualize multiscale data, ranging from the micro-scale to the geographic scale. This platform not only enhances data interoperability but also facilitates a comprehensive and integrated understanding of geological and geographical dynamics, prioritizing effective communication to convey information to end users. These features make this web solution a valuable tool for territorial planning, resource management, and the study of interactions across various spatial scales.

**Author Contributions** Gaetano Ortolano: conceptualization, original draft, methodology, writing review and editing. Gianfranco Di Pietro and Alberto D'Agostino: methodology, software development, writing review and editing, validation. Eugenio Fazio: investigation, writing review and editing, validation, funding administration. Rosaria Ester Musumeci and Rosolino Cirrincione: validation, review, project administration.

**Funding** Open access funding provided by Università degli Studi di Catania within the CRUI-CARE Agreement. Part of this work was developed under funding by University of Catania - PIA.CE.RI. 2024-2026, project: GEO-ROUTES n. 22722132195; Scientific Coordinator: Eugenio Fazio.

**Data Availability** All code, library and Jupyter Notebooks are published in the following Github respository: https://github.com/gianfrancodp/MGStools. A running live webview of the MGS build for the case history is available at the website: https://gianfrancodp.github.io/metpet.

## Declarations

**Competing interests** The authors declare no competing interests.



## References

Acevedo Zamora MA, Schrank CE, Kamber BS (2024) Using the traditional microscope for mineral grain orientation determination: A prototype image analysis pipeline for optic-axis mapping (POAM). J Microsc 295(2):147–176. https://doi.org/10.1111/jmi.13284

Arnaud R, Barnes MC (2006) COLLADA: Sailing the gulf of 3D digital content creation. A K Peters, Wellesley, Mass

Autant-Bernard C, Fadairo M, Massard N (2013) Knowledge diffusion and innovation policies within the European regions: Challenges based on recent empirical evidence. Res Policy 42(1):196–210. https://doi.org/10.1016/j.respol.2012.07.009

Azmi NA, Shafri HZM, Abidin FAZ, Shaharum NSN, Al-Habshi MMA (2022) Development of WebGIS using open source geospatial technologies for Krau Wildlife Reserve. IOP Conf Ser Earth Environ Sci 1064(1):012016. https://doi.org/10.1088/1755-1315/1064/1/012016

Bachri S, Sumarmi, Irawan LY, Utaya S, Wirawan R, Nurdiansyah FD, Nurjanah AE, Tyas LW, Adillah AA, Setia D (2022) FOSS (Free Open Source Software) integration to implement WebGIS-based information system of kelud volcano. IOP Conf Ser Earth Environ Sci 1066(1):012010. https://doi.org/10.1088/1755-1315/1066/1/012010

Balla D, Gede M (2024) Beautiful thematic maps in Leaflet with automatic data classification. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLVIII-4/W12-2024, 3–10. https://doi.org/10.5194/isprs-archives-XLVIII-4-W12-2024-3-2024

Baru C, Lin K (2009) Mediating among GeoSciML resources. Int J Digit Earth 2(sup1):18–28. https://doi.org/10.1080/17538940902912437

Baumann P, Misev D, Merticariu V, Huu BP (2021) Array databases: Concepts, standards, implementations. J Big Data 8(1):28. https://doi.org/10.1186/s40537-020-00399-2

Burggraf D (2015) Category: OGC® Implementation

Caso F, Piloni C, Filippi M, Pezzotta A, Fazio E, Visalli R, Zucali M (2024) Combining traditional and quantitative multiscale structural analysis to reconstruct the tectono-metamorphic evolution of migmatitic basements: The case of the Valpelline Series, Dent-Blanche Tectonic System, Western Alps. J Struct Geol 182:105099. https://doi.org/10.1016/j.jsg.2024.105099

Cheng D (2024) Research on HTML5 responsive web front-end development based on bootstrap framework. In: 2024 7th International conference on computer information science and application technology (CISAT) pp 711–718. Hangzhou, China: IEEE

Cirrincione R, Fazio E, Fiannacca P, Ortolano G, Pezzino A, Punturo R (2015) The Calabria-Peloritani Orogen, a composite terrane in Central Mediterranean; its overall architecture and geodynamic significance for a pre-Alpine scenario around the Tethyan basin. Periodico di Mineralogia 84(3B):701–749. https://doi.org/10.2451/2015PM0446

Danchilla B (2012) Three.js Framework. In Beginning WebGL for HTML5 pp 173–203. Berkeley, CA: Apress

Duarte L, Queirós C, Teodoro AC (2021) Comparative analysis of QGIS plugins for Web Maps creation. La Granja 34(2):8–26. https://doi.org/10.17163/lgr.n34.2021.01

Elrom E (2016) CSS, Bootstrap, & Responsive Design. Pro MEAN Stack Development. Apress, Berkeley, CA, pp 131–164

Fazio E, Ortolano G, Alsop G, D'Agostino A, Visalli R, Luzin V, Cirrincione R (2024) Enhanced structural analysis through a hybrid analogue-digital mapping approach: Integrating field and UAV survey with microtomography to characterize metamorphic rocks. J Struct Geol 187:105213. https://doi.org/10.1016/j.jsg.2024.105213

Foerster T, Stoter J, Van Oosterom P (2012) On-demand base maps on the web generalized according to user profiles. Int J Geogr Inf Sci 26(1):99–121. https://doi.org/10.1080/13658816.2011.574292

Fossen H (2016) Structural geology (2nd Ed). Cambridge University Press

Gede M (2023) Automatic Labels in Leaflet. Adv Cartogr GIScience ICA 4:1–5. https://doi.org/10.5194/ica-adv-4-8-2023

González Canché MS (2023) Data formats, coordinate reference systems, and differential privacy frameworks. Spatial socio-econometric modeling (SSEM). Springer International Publishing, Cham, pp 55–94

Han Y (2015) Cloud storage for digital preservation: Optimal uses of Amazon S3 and Glacier. Library Hi Tech 33(2):261–271. https://doi.org/10.1108/LHT-12-2014-0118

Heilbronner R (2000) Automatic grain boundary detection and grain size analysis using polarization micrographs or orientation images. J Struct Geol 22(7):969–981. https://doi.org/10.1016/S0191-8141(00)00014-6

Hossain M (2014) CORS in action: Creating and consuming cross-origin APIs. New York: Manning Publications Co. LLC. (Description based on publisher supplied metadata and other sources)

Krygier J, Wood D (2025) Making maps: A visual guide to map design for GIS. Guilford Press, New York

Kumari P, Kumari M, Azmat A, Kumari A (2023) Exploring static website development a fundamental analysis of design and functionality. Int J Commun Syst Netw Technol 11. https://doi.org/10.18486/ijcsnt.2023.11.2.08

Li N, Zhang B (2021) The Research on Single Page Application Front-end development Based on Vue. J Phys: Conf Ser 1883(1):012030. https://doi.org/10.1088/1742-6596/1883/1/012030

Li Y, Onasch CM, Guo Y (2008) GIS-based detection of grain boundaries. J Struct Geol 30(4):431–443. https://doi.org/10.1016/j.jsg.2007.12.007

Lombardo V, Piana F, Mimmo D (2018) Semantics-informed geological maps: Conceptual modeling and knowledge encoding. Comput Geosci 116:12–22. https://doi.org/10.1016/j.cageo.2018.04.001

Maestri L (2017) Visualization of computer-generated 3D cities using GIS data p 66. https://doi.org/10.34726/HSS.2017.29835

Maguire D, Longley P (2005) The emergence of geoportals and their role in spatial data infrastructures. Comput Environ Urban Syst 29(1):3–14. https://doi.org/10.1016/S0198-9715(04)00045-6

Mamtani MA (2025) The future of structural geology in the 21st century - moving from mesoscale to nanoscale observations in tectonically deformed rocks. J Geol Soc India 101(1):10–23. https://doi.org/10.17491/jgsi/2025/174056

Markieta M, Rinner C (2014) Using distributed map overlay and layer opacity for visual multi-criteria analysis. Geomatica 68(2):95–105. https://doi.org/10.5623/cig2014-202

Mason P (2020) JavaScript Libraries. SAS Stored Processes. Apress, Berkeley, CA, pp 81–97

McInerney D, Kempeneers P (2015) Image overviews, tiling and pyramids. Open source geospatial tools. Springer International Publishing, Cham, pp 85–97

Melsom J (2020) Multi-scalar Geo-landscape Models: Interfacing Geological Models with Landscape Surface Data. Wichmann Verlag, DE

Mete MO, Yomralioglu T (2021) Implementation of serverless cloud GIS platform for land valuation. Int J Digital Earth 14(7):836–850. https://doi.org/10.1080/17538947.2021.1889056

Miles V, Esau I, Pettersson L (2024) Using web GISto promote stakeholder understanding of scientific results in sustainable urban development: A case study in Bergen. Norway Sustain Dev 32(3):2517–2529. https://doi.org/10.1002/sd.2787

Naik PG, Naik GR (2024). Mastering bootstrap, AJAX, and jQuery for elevating web experiences with advanced development techniques (Covers DOM Manipulation). Shashwat Publication

Norman DA (2013) The design of everyday things (Revised and expanded, edition. Basic Books, New York, New York

Ortolano G, Cirrincione R, Pezzino A, Puliatti G (2013) Geo-Petro-Structural study of the Palmi shear zone: Kinematic and rheological implications. Rendiconti Online della Società Geologica Italiana 29:125–129

Ortolano G, Zappalà L, Mazzoleni P (2014) X-Ray Map Analyser: A new ArcGIS® based tool for the quantitative statistical data handling of X-ray maps (Geo- and material-science applications). Comput Geosci 72:49–64. https://doi.org/10.1016/j.cageo.2014.07.006

Ortolano G, Visalli R, Godard G, Cirrincione R (2018) Quantitative X-ray Map Analyser (Q-XRMA): A new GIS-based statistical approach to mineral image analysis. Comput Geosci 115:56–65. https://doi.org/10.1016/j.cageo.2018.03.001

Ortolano G, Fazio E, Visalli R, Alsop GI, Pagano M, Cirrincione R (2020) Quantitative microstructural analysis of mylonites formed during Alpine tectonics in the western Mediterranean realm. J Struct Geol 131:103956. https://doi.org/10.1016/j.jsg.2019.103956

Ortolano G, D'Agostino A, Pagano M, Visalli R, Zucali M, Fazio E, Cirrincione R (2021) ArcStereoNet: A New ArcGIS® toolbox for projection and analysis of meso- and micro-structural data. ISPRS Int J Geo Inf 10(2):50. https://doi.org/10.3390/ijgi10020050

Passchier CW, Trouw RAJ (2005) Microtectonics (1. Aufl, Online-Ausg Ed). Berlin [u.a.]: Springer

Passchier CW, Trouw RAJ (1998) A framework of microtectonic studies. Microtectonics. Berlin, Heidelberg, Springer, Berlin Heidelberg, pp 1–6

Preda M, Arsov I, Moran F (2010) COLLADA + MPEG-4 OR X3D + MPEG-4. IEEE Veh Technol Mag 5(1):39–47. https://doi.org/10.1109/MVT.2009.935544

Prosser G, Caggianelli A, Rottura A, Del Moro A (2003) Strain localisation driven by marble layers: The Palmi shear zone (Calabria-Peloritani terrane, Southern Italy). GeoActa 2:155–156. arXiv:11563/4086

Rahman A, Cahyono A (2023) Analysis Of 3-D Building Modeling Using Photogrammetric Software: Agisoft Metashape And Micmac. IOP Conf Ser Earth Environ Sci 1276(1):012044. https://doi.org/10.1088/1755-1315/1276/1/012044

Roda M, Zucali M, Corti L, Visalli R, Ortolano G, Spalla MI (2021) Blueschist mylonitic zones accommodating syn-subduction exhumation of deeply buried continental crust: The example of the Rocca Canavese Thrust Sheets Unit (Sesia-Lanzo Zone, Italian Western Alps). Swiss J Geosci 114(1):6. https://doi.org/10.1186/s00015-021-00385-7

Romero-Organvidez D, Horcas J- M, Galindo JA, Benavides D (2024) Data visualization guidance using a software product line approach. J Syst Softw 213:112029. https://doi.org/10.1016/j.jss.2024.112029

Russo D, Fiannacca P, Fazio E, Cirrincione R, Mamtani MA (2023) From floor to roof of a batholith: Geology and petrography of the north-eastern Serre Batholith (Calabria, southern Italy). J Maps 19(1):2149358. https://doi.org/10.1080/17445647.2022.2149358

Simon M (2023) Manipulating HTML Elements. JavaScript for web developers. Apress, Berkeley, CA, pp 93–120

Snyder JP (1993) Flattening the earth: Two thousand years of map projections. University of Chicago Press, Chicago

Steiniger S, Hunter AJ (2013) The 2012 free and open source GIS software map - A guide to facilitate research, development, and adoption. Comput Environ Urban Syst 39:136–150. https://doi.org/10.1016/j.compenvurbsys.2012.10.003

Tarquini S, Armienti P (2011) Quick determination of crystal size distributions of rocks by means of a color scanner. Image Anal Stereol 22(1):27. https://doi.org/10.5566/ias.v22.p27-34

Tarquini S, Favalli M (2010) A microscopic information system (MIS) for petrographic analysis. Comput Geosci 36(5):665–674. https://doi.org/10.1016/j.cageo.2009.09.017

Tavani S, Corradetti A, Rizzo R, Seers T (2024) Best practices towards the digitization of 3D traces from virtual outcrop models. J Struct Geol 186:105222. https://doi.org/10.1016/j.jsg.2024.105222

Teodorovici VG (2013) jQuery, jQuery UI and jQuery Mobile: Recipes and examples by Adriaan de Jonge and Phil Dutson. ACM SIGSOFT Softw Eng Notes 38(5):68–69. https://doi.org/10.1145/2507288.2507294

Thiele ST, Grose L, Samsu A, Micklethwaite S, Vollgger SA, Cruden AR (2017) Rapid, semi-automatic fracture and contact mapping for point clouds, images and geophysical data. Solid Earth 8(6):1241–1253. https://doi.org/10.5194/se-8-1241-2017

Vakali A, Pallis G (2003) Content delivery networks: Status and trends. IEEE Internet Comput 7(6):68–74. https://doi.org/10.1109/MIC.2003.1250586

Vasuki Y, Holden E-J, Kovesi P, Micklethwaite S (2014) Semi-automatic mapping of geological Structures using UAV-based photogrammetric data: An image analysis approach. Comput Geosci 69:22–32. https://doi.org/10.1016/j.cageo.2014.04.012

Viglino J-M (2024). Ol-ext: Extensions for OpenLayers

Visalli R, Ortolano G, Godard G, Cirrincione R (2021) Micro-fabric analyzer (MFA): A new semiautomated ArcGIS-based edge detector for quantitative microstructural analysis of rock thin-sections. ISPRS Int J Geo Inf 10(2):51. https://doi.org/10.3390/ijgi10020051

Visser M, Sikkenga B, Berry M (2018) Digital marketing fundamentals: From strategy to ROI (First edition Ed). Groningen/Utrecht, The Netherlands: Noordhoff Uitgevers B.V. (Marjolein Visser is Managing Consultant of MarketWise and founder of TourWise. Berend Sikkenga is Head of eCommerce at LEGO. Mike Berry is a Digital Marketing consultant, trainer and Professor of Practice at Hult International Business School, USA)

Walny J, Frisson C, West M, Kosminsky D, Knudsen S, Carpendale S, Willett W (2020) Data changes everything: Challenges and opportunities in data visualization design handoff. IEEE Trans Visual Comput Graphics 26(1):12–22. https://doi.org/10.1109/TVCG.2019.2934538

Wei Y, Yuan H, Li H (2024) Exploring the contribution of advanced systems in smart city development for the regeneration of urban industrial heritage. Buildings 14(3):583. https://doi.org/10.3390/buildings14030583

Wise DU, Dunn DE, Engelder JT, Geiser PA, Hatcher RD, Kish SA, Schamel S (1984) Fault-related rocks: Suggestions for terminology. Geology 12(7):391. https://doi.org/10.1130/0091-7613(1984)12<391:FRSFT>2.0.CO;2

Wohlmann J (2024) Expanding the field of view - a simple approach for interactive visualisation of electron microscopy data. J Cell Sci 137(20):jcs262198. https://doi.org/10.1242/jcs.262198

Xypolias P (2010) Vorticity analysis in shear zones: A review of methods and applications. J Struct Geol 32(12):2072–2092. https://doi.org/10.1016/j.jsg.2010.08.009