AI4PHENO
AWARENESS BLOOMING

# Project Deliverables Verification Document (AI4Pheno)

Version 1.0.0

Seth Software sp. z o.o
Poznań University of Life Sciences
September 15, 2023

# Contents

1 About project 2

1.1 Web Application Access . . . 2

1.2 Problem Definition . . . 2

1.3 User Benefits . . . 2

1.4 Objectives . . . 3

2 Compliance with Software Requirements 3

2.1 Functional requirements . . . 3

2.2 Performance requirements . . . 13

2.3 Platform and infrastructure requirements . . . 14

2.4 Support and maintenance requirements . . . 15

3 Compliance with Methodology and organization of the work and resource 16

3.1 Methodology . . . 16

3.2 Organization of work and resource . . . 23

3.3 Quality control measures . . . 24

4 Contact 27

# 1 About project

This document, *"Project Deliverables Verification Document (AI4Pheno)"*, serves as confirmation that the tender has been executed in accordance with the specification attached to the tender found at https://www.egi.eu/tender-eosc-dih-2023.

The specifications to which this document refers can be found at the following links:

- Tender Specifications
- EGI-2023-001 - Annex 8.1 - Technical offer part I
- EGI-2023-001 - Annex 8.2 - Technical offer part II

## 1.1 Web Application Access

The web application is available and can be accessed at the following URL: `https://ai4pheno.seth.software`.

## 1.2 Problem Definition

The heart of the problem lies in the domain of *Phenology* - a discipline studying vegetation's temporal changes. Traditionally, it targets seasonal events such as budding, fruiting, flowering, and ageing. Due to the influence of both environmental factors and human activities on plant developmental stages, this science yields invaluable insights into the state of our landscape's vegetation cover. Such data proves pivotal especially in the context of climate change, where progressive shifts challenge the health and occurrence of plant species.

While various methods exist to detect phenological stages, there remains a void in the market for open tools aiding in the collection and analysis of digital phenological imagery using machine learning (ML) techniques. The proposed tool intends to fill this gap by offering an automated, efficient means to apply ML in time series image analyses of vegetation impacted by climate changes.

## 1.3 User Benefits

Upon the provision of this service, users will be endowed with a comprehensive IT solution. This integrated platform will not only streamline the processes of image acquisition, storage, and analysis but also offer scalability to accommodate further research areas like landscape analysis or crop yield forecasting.

### 1.4 Objectives

The overarching aim is to co-develop this solution with the EOSC DIH and the Research Community, ensuring its accessibility via the EOSC Marketplace. By procuring these solutions from the private sector, EOSC DIH aims to meet the research community's needs while also enriching the EOSC with novel offerings.

Specific objectives encompass:

1. Acquisition of a digital platform for phenological imagery.
2. Provision of digital tools for manual and automatic image analysis.
3. Introduction of digital tools to utilize hand crafted AI models for phenological imagery analysis.
4. Development of AI models for automatic ROI detection.
5. Facilitation of management tools for the platform concerning data and users.

## 2 Compliance with Software Requirements

### 2.1 Functional requirements

Table 1: Functional requirements

| ID | Type | Description according to technical specifications |
|---|---|---|
| **GFR_1** | System | The whole system (web application) was shared as open source software under the Apache License 2.0. This meant that the entire source code could be used, modified, and redistributed without any issues. Source code and appropriate documentation were shared at https://github.com/EOSC-AI4PHENO/AI4PhenoEOSC.git. |
| **GFR_2** | System | The whole system was developed inhouse and was hosted using an on-premises approach. The web application was developed in the JAVA language. The deployment was prepared for the Ubuntu platform to avoid any additional costs and to be in compliance with the Apache license 2.0. Additionally, the AI models were implemented in the Python language using the TensorFlow and PyTorch frameworks. |
| **GFR_3** | System | The entire infrastructure on premises was designed for high availability. Furthermore, the web application was deployed on high availability (HA) services. This meant that the full environment, including the application layer, was fault tolerant with no service interruptions. All the aforementioned assumptions allowed the system to operate in the Market Place EOSC for a minimum of one year from delivery. |

Continued on the next page

Table 1 Functional requirements –Continued on the next page

| ID | Type | Description according to technical specifications |
|---|---|---|
| **GFR_4** | System | The web application was compatible with most of the then-current version browsers for Windows, Mac, Android, and iOS systems. The application was written in React, communicating through an API backend, which then interfaced with the AI inference server. |
| **GFR_5** | Users | In the system, min. 3 roles were developed:<br>1. Administrators<br>2. Power users – users of cameras and analysis tools<br>3. Regular Users - mainly users in read-only mode<br>Based on the roles created, 3 types of users were introduced. The power user was the most frequently used. |
| **GFR_6** | Users | During the sign-up functionality, each user had created a detailed profile in which she/he had provided at least the following information for statistical purposes:<br>• Organisation<br>• Country of organisation<br>• Position/level of expertise<br>• Scope of activity<br>• Purpose of use<br>• Project/funder |

Continued on the next page

Table 1 Functional requirements –Continued on the next page

| ID | Type | Description according to technical specifications |
|---|---|---|
| **GFR_7** | Admin | The standard functionality of accounts and profiles in the system was implemented. |
| **SFR_DAS_1** | Camera data | Two data sources are used in the system. The data is collected from two types of cameras: Dahua 5mpx and Dahua 12mpx. Both types of cameras contained configurable image stream schedules and had the possibility to download the image database at the scheduled time with defined frequencies every 30 minutes.<br>The communication functionality was configured in the system independently based on the logged-in administrator role. |
| **SFR_DAS_2** | Camera data | Besides the two types of cameras mentioned in point `SFR_DAS_1`, additional data sources are being used from on premises, located on the tenderer side. The functionality of this additional camera/data source is similar, but it allows for the proper verification of services and functionalities using devices located at their own premises. |
| **SFR_DAS_3** | Camera data | All cameras that are used in the project had the functionality to transfer data/images from an SD Card (if present). This functionality allowed for the downloading of images on demand directly saved on the SD card. |

Continued on the next page

Table 1 Functional requirements –Continued on the next page

| ID | Type | Description according to technical specifications |
|---|---|---|
| **SFR_DAS_4** | Input data | Given the resolution of camera type 1 (12MPx) and type 2 (5MPx), the narrowness of mobile bandwidth (sim card), and the bit depth required for a colorful image, the most appropriate image standard is JPG/JPEG. |
| **SFR_DAS_5** | Camera data | The standard functionality currently is implemented allows for scheduled image data downloads from the camera based on predefined configurations (e.g., at a frequency of every 30 minutes).<br>In the event of communication issues with the cameras, users have the option to directly download image data from the SD Card. |
| **SFR_DAS_6** | Camera data | Depending on the camera type, the system generates alerts for malfunctions. |
| **SFR_DAS_7** | Camera auxiliary data | Apart from the desired data (image data) can be stored for cameras, additional data can be stored in the system. |

Continued on the next page

Table 1 Functional requirements –Continued on the next page

| ID | Type | Description according to technical specifications |
|---|---|---|
| **SFR_DAS_8** | Camera auxiliary data | Based on the provided auxiliary data, we can:<br>1. Manage<br>2. Shape<br>3. Clean<br>4. Filter<br>5. Upload historical and current data in the form of images<br>6. Download time series to `.xlsx` |
| **SFR_DAS_9** | Third-party data sources | The system currently retrieves meteorological data from open/public meteorological databases. A primary example of such a database is the EDWIN Meteo API. |
| **SFR_DAS_10** | Image data | Currently, the tool offers the following set of functionalities:<br>- Load image.<br>- Zoom in and out.<br>- Select all ROI.<br>- Delete selected ROI.<br>- Choose region shapes which include: rectangle, circular, elliptical, polygon.<br>- Save to a json file. |

Continued on the next page

Table 1 Functional requirements –Continued on the next page

| ID | Type | Description according to technical specifications |
|---|---|---|
| **SFR_DAS_11** | Image data | The image source searching module in the system is implemented with criteria such as camera type, species, etc. |
| **SFR_DAS_12** | Image data | Image selection is currently carried out to eliminate images that are either too bright or too dark. This is achieved by making decision images based on the position of the sun relative to the camera location and histogram algorithm. |
| **SFR_DA_1** | ROI definition | The tool for manual ROI (mROI) definition is currently available in the system. This tool allows users to define and modify the vertices of the polygon. |
| **SFR_DA_2** | ROI definition | The tool for manual ROI (Region of Interest) definition currently allows users to create more than one ROI in the reference image, referred to as multi manual ROI (mmROI). |
| **SFR_DA_3** | ROI definition | After loading the raster image, there is a functionality (represented by a dedicated button on the web page) that invokes automatic detection of ROI (aROIs) using the state-of-the-art machine learning algorithm Mask-RCNN for Linden and Apple trees. |
| **SFR_DA_4** | ROI definition | Uploading any pre-trained model (specifically for automatic definition of ROIs) into the system or service must be compatible with the *MaskR-CNN* model and requires support from the system administrator. |
| **SFR_DA_5** | ROI data | For segmented ROIs, functionality of ROI analytics data will be provided: RGB indexes, height, width, area. |

Continued on the next page

Table 1 Functional requirements –Continued on the next page

| ID | Type | Description according to technical specifications |
|---|---|---|
| **SFR_DA_6** | ROI data | For segmented ROIs, the functionality provides additional analytics data for the ROI (red DN and green DN, red DN and blue DN, green DN and blue DN) |
| **SFR_DA_7** | ROI data averaged | In the current approach, the computation of red, green, and blue digital numbers (DN) is carried out over the ROI. The DN values for each color component range from 0 to 255. |
| **SFR_DA_8** | ROI data averaged fitting | In the current setup, we are fitting a vegetation curve to the ROI data. |
| **SFR_DA_9** | Image ROI meta data | Time and data retrieval from file properties or/and images (via date stamp) or/and filenames is currently possible in the system. |
| **SFR_DA_10** | Image and ROI data custom | Customised colour indexes (CIs) can be uploaded with administrator support. |
| **SFR_AIM_1** | Apple AI model | Based on the training images provided, a ready-to-use AI model for automatic ROIs semantic segmentation of apple fruit is developed and deployed in service. The desired quality for the model is given by *mIoU* $> 0.75$. In our approach, we achieve *mIoU* $= 79.19$. |

Continued on the next page

Table 1 Functional requirements –Continued on the next page

| ID | Type | Description according to technical specifications |
|---|---|---|
| **SFR_AIM_2** | Flowering<br>AI model | On the basis of the training images provided, a ready-to-use AI model for automatic ROIs semantic segmentation for the flowering stage of the European linden is being developed and deployed. The desired quality of the model is specified as: mIoU $> 0.70$. In our approach, we achieve: mIoU $= 85.21$. |
| **SFR_DV_1** | Image & ROI | Currently, the platform provides functionality to showcase data in a comprehensive manner. Users can:<br>• **Print ROI on the Selected Image:** The Region of Interest (ROI) is dynamically highlighted on the chosen image, allowing for an immediate visual analysis.<br>• **Export Results to JSON Format:** After the analysis, users have the option to export their results directly into a JSON format, facilitating easy data sharing and further processing. |
| **SFR_DV_2** | Image & ROI | A screen is prepared that presents the captured images in the form of a list with photo thumbnails. The list can be filtered by camera, species, and date range. Upon selecting an item from the list, a full-size image is displayed with defined ROIs marked on the photo. |

Continued on the next page

Table 1 Functional requirements –Continued on the next page

| ID | Type | Description according to technical specifications |
|---|---|---|
| **SFR_DV_3** | Image & ROI | Currently, we visualize the quantity and characteristics of the identified objects. This encompasses various presentation methods such as graphs and statistical analyses. |
| **SFR_DV_4** | ROI data | Currently, we are visualizing the results of the Region of Interest (ROI) analysis over time. One of the primary methods applied for this visualization is the use of graphs. These graphical representations provide an intuitive insight into the temporal progression of the ROI data. |
| **SFR_DV_5** | ROI data | In the present context, visualizing vegetation curves becomes pivotal for numerous ecological and environmental studies. By using the *Region of Interest (ROI)-averaged approach*, one can derive significant insights from such visualizations. |
| **SFR_DV_6** | ROI data | Various types of charts are implemented in the system, and it allows for the overlay of different analyses. |
| **SFR_DV_7** | ROI data | Visualization of images, where defined objects (Linden and Apple) are identified by AI algorithm, is available in the system. |
| **SFR_DV_8** | ROI data | Visualization of apples identification results provided by the AI algorithm are available in the system. |
| **SFR_DV_9** | Camera data | Visualization of camera data in the form of images is available in the system. |

Continued on the next page

Table 1 Functional requirements –Continued on the next page

| **ID** | **Type** | **Description according to technical specifications** |
|---|---|---|
| **SFR_DV_10** | Camera data | Based on the location data of the camera, a screen is prepared for presenting the location of devices using Google Maps. Locations are visible on the screen in the form of “pin” markers with descriptions displayed after hovering over the object. Depending on the presentation area, Google Maps displays the location data of devices within the presentation area. |
| **SFR_DV_11** | User data | A list of all users/user account data can be displayed in the system. |
| **SFR_DV_12** | Admin data | A list of all additional information for users/user account data can be displayed in the system (for admin account). |
| **SFR_DE_1** | Image data | The system allows for the export of images to local media and also provides the capability to export to Google Drive. |
| **SFR_DE_2** | Analysis data | The system allows for the export of analysis results to local media. |
| **SFR_DE_3** | ROI data | The system currently offers the functionality to *export ROI data (polygon coordinates) to a JSON file*. |

## 2.2 Performance requirements

Table 2: Performance requirements

| ID | Performance requirements |
|---|---|
| PR_1 | • **Service Level Agreement (SLA)**: The current Service Level Agreement (SLA level) stands at 99.50%. This indicates that the allowed periods of downtime/unavailability are:<br>• Daily: less than 7m 12s<br>• Weekly: less than 50m 24s<br>• **Data Durability**: The data's (including the system) present durability stands at a 99.97% level. This relates to the ongoing persistence of data. We achieve this by enhancing the durability of both the data (including source code) and the storage infrastructure. A high durability level guarantees protection against bit rot, degradation, and any type of data corruption or loss.<br>• **Response Time**: A traditional or regular webpage request responds within 1000 milliseconds. However, some functionalities, particularly those based on machine learning algorithms, may have longer response times due to the algorithms' complexity.<br>• **Batch Processing**: Any potential batch processing completes within 60 minutes.<br>• **System Monitoring**: We currently monitor all subsystems of the project using phpservermon (PHP Server Monitor). It logs any availability issues to a MySQL database. |

## 2.3 Platform and infrastructure requirements

Table 3: Platform and infrastructure requirements

| ID | Platform and infrastructure requirements |
|---|---|
| **PIR_1** | The entire infrastructure is on-premises (non-cloud solution). For servers (primarily Dell servers), we apply a high availability approach based on virtualization/containerization. Both the web application and REST API solutions are exposed by at least 2 nodes (web servers), with a load balancer in place to distribute application traffic across the available servers. We use load balancers to enhance capacity (in terms of concurrent users) and to ensure the reliability/availability of the applications (both the web application and REST API).<br>Machine learning models are provided by a Python environment based on the tensorflow/pytorch framework and are exposed to a web form application developed in Java.<br>The entire source code, documentation, and any required documents are placed in a repository system with public access, exposed under the Apache License 2.0.<br>Furthermore, the AI server appies Rabbitmq for queuing, distributing tasks across cores using Celery. Inferences are executed through Nvidia Triton, which then returns the result to Redis. |

## 2.4 Support and maintenance requirements

Table 4: Support and maintenance requirements

| ID | Support and maintenance requirements |
|---|---|
| **SPPM_1** | The documentation/tutorials are provided in the English language and naturally include information on 1) camera connection, 2) data processing and analysis, as the service caters to the needs of the agriculture research community.<br>Furthermore, a dedicated customer support system (https://glpi-project.org/ ticket system) is available for managing user tickets related to the web application. The ticket system is currently available only for internal purposes. |

# 3 Compliance with Methodology and organization of the work and resource

## 3.1 Methodology

Table 5: Methodology

| **Activities or tasks to be carried out** | **Methodology to be implemented.** |
|---|---|
| **A1 Development of tools for data transfer and storage** | Depending on the chosen camera type, company, or platform, different approaches to image/data acquisition are considered. In this project, by default, two camera types are taken into account:<br>1. Camera type 1 (low resolution) - Dahua IP 5Mpx, Dahua Technology, China.<br>2. Camera type 2 (high resolution) - 12 MPx Dahua, Dahua Technology, China. |

Continued on the next page

Table 5 Methodology –Continued on the next page

| **Activities or tasks to be carried out** | **Methodology to be implemented.** |
|---|---|
| **A1.1 Camera data transfer type 1** | This type of camera *is* a standard IP camera with direct access to image/data based on the camera's IP settings.<br>Most requests to this type of camera are based on a standard GET Request with parameters in the form of a query string. If the request is successful, the IP camera returns an HTTP header with a status of `200 OK`. The HTTP Body response contains the actual data or an error message if an error occurs.<br>The communication format *is* based on a standard GET Form, and the response comes in the form of `text/plain`.<br>The standard functionalities for this type of IP camera include:<br>1. Management of configuration and viewing camera settings<br>2. Management of event handlers<br>3. Management of alarms<br>4. Management of records and snaps<br>5. Retrieval of the current image/snap<br>6. Retrieval of an image/snap from the SD card |

Continued on the next page

Table 5 Methodology –Continued on the next page

| **Activities or tasks to be carried out** | **Methodology to be implemented.** |
|---|---|
| **A1.2 Camera data transfer type 2** | This type of camera *is* a standard IP camera with direct access to image/data based on the camera's IP settings.<br>Most requests to this type of camera are based on a standard GET Request with parameters in the form of a query string. If the request is successful, the IP camera returns an HTTP header with a status of `200 OK`. The HTTP Body response contains the actual data or an error message if an error occurs.<br>The communication format *is* based on a standard GET Form, and the response comes in the form of `text/plain`.<br>The standard functionalities for this type of IP camera include:<br>1. Management of configuration and viewing camera settings<br>2. Management of event handlers<br>3. Management of alarms<br>4. Management of records and snaps<br>5. Retrieval of the current image/snap<br>6. Retrieval of an image/snap from the SD card |

Continued on the next page

Table 5 Methodology –Continued on the next page

| **Activities or tasks to be carried out** | **Methodology to be implemented.** |
|---|---|
| **A1.3 Image database** | All data *is* stored in a PostgreSQL engine, supported with high availability features.<br>To facilitate the maintenance of a large database and ensure consistency, the original images are stored on system Fedora. This approach prevents discrepancies and mitigates potential issues. Additional information, such as ROI and labels, is also stored in the database in formats like CSV, JSON, and so on.<br>In addition to the standard functionalities, the system also includes the following features:<br>1. Over or underexposed image detection.<br>2. Anomaly detection in images.<br>Information stored, including paths to the files, coordinates of ROI, segmented areas, and annotated labels, *is* provided through the implemented methods. Data served in this manner *is* suitable for training, validating, and testing the model. |
| **A1.4 Backend for transfer and storage management v.1** | The main web application Was developed in the Java language, which implements API calls for camera type 1 and type 2. For different camera types and specifications, various image resolutions are supported. Original image files are stored in the Fedora system. |
| **A1.5 Manual (v1 version) for integration of type 1 and type 2 cameras into the system** | Based on the implemented API calls for at least two camera types, detailed documentation is provided in the user guide. |

Continued on the next page

Table 5 Methodology –Continued on the next page

| Activities or tasks to be carried out | Methodology to be implemented. |
|---|---|
| **A2 Development of the tools for data processing and visualization in GUI** | It is assumed that the web application is developed in React (frontend) and is responsible for data presentation and visualization. Meanwhile, all data processing and potential image generation are performed on the Python backend side. An open-source library for data presentation and visualization in Java is proposed.<br>According to the requirements, the system includes the use of automatic image preprocessing operations such as:<br>1. Histogram equalization,<br>2. Contrast, brightness, and sharpness enhancement,<br>3. Deblurring, filtering/denoising (median, gaussian, etc).<br>For each stored image, the developed tools display:<br>1. The original image,<br>2. Marked areas annotation, GT masks (if any),<br>3. Image segmentation and detection results (from machine learning models). |

Continued on the next page

Table 5 Methodology –Continued on the next page

| **Activities or tasks to be carried out** | **Methodology to be implemented.** |
|---|---|
| **A2.1 Deployment of the final solution to production environment** | The project source code is placed in a repository system based on GITHUB, accessible at: `https://github.com/EOSC-AI4PHENO/AI4PhenoEOSC.git`. This accelerates the deployment of the current and final solution. Suitable CI/CD - Continuous Integration (CI) and Continuous Delivery (CD) automation is considered. The final machine learning models are saved in their native corresponding format. There is also the option to store these models on GitLab LFS (Large File Storage) repository. |
| **A3 Development of models for phenological and AI analysis** | In general, machine learning models *were* prototyped using the `Tensorflow` and `Pytorch` frameworks in the `Python` language. These models *were* then exposed for a web application based on the `RESTAPI` approach, specifically using `FastAPI`.<br>The AI server consisted of a combination of technologies: `FastAPI`, `Rabbimq`, `Celery`, `Nvidia Triton`, `Redis`, and `flower`. |
| **A3.1 Collection of data for modelling** | The acquisition of data for modelling was one of the most crucial tasks in machine learning model development. Especially in supervised models, a proper ground truth was necessary for the collected data (images). This meant that it was required to extract ROI from images either manually or in an automated manner, as well as other information (label, segmented mask). To create the ground truth, the labeling process had been applied, which allowed marking of objects by various methods: polygon, circle, rectangular, elliptical, and polyline region shapes, labeling, etc. |

Continued on the next page

Table 5 Methodology –Continued on the next page

| **Activities or tasks to be carried out** | **Methodology to be implemented.** |
|---|---|
| **A3.2 Development of models in laboratory version** | The first step in machine learning development was exploratory data analysis. This step took approximately 30% of the total ML development. This step was the most important because the result of that process had a direct impact on the quality of the input data used to train the model. At this stage, the calibration of image preprocessing took a special place.<br>Generally in this project, deep learning image segmentation/classification algorithms were mainly used. In this area, the most efficient network architectures (based on CNN networks) has been found: Mask RCNN which is state-of-the-art algorithm.<br>To calculate the predictive power of machine learning models, the cross-validation technique *should have been* applied. As a good practice in machine modeling, the whole dataset (numerical features extracted/generated based on ROIs) *should have been* divided into 3 subsets:<br>1. Training subset<br>2. Validation subset<br>3. Test subset<br>In this project, the method of cross-validation that was applied.<br>At the end of every cross-validation process, computation of model performance *was* performed in the form of well-known metrics. The model performance metrics *decided* on the final model selection. Depending on which machine learning algorithm *was* chosen, the metric which *could be* applied *were* the following:<br>1. Metric for classification, based on confusion matrix:<br>22 (a) accuracy<br>2. Metric for object detection/segmentation:<br>(a) MIoU for segmentation |

Continued on the next page

Table 5 Methodology –Continued on the next page

| **Activities or tasks to be carried out** | **Methodology to be implemented.** |
|---|---|
| **A3.3 Development of models in prototype version** | The saved models were finally deployed to production. They operated in inference mode, utilizing technologies such as *FastAPI*, *RabbitMQ*, *Celery*, *NVIDIA Triton*, *Flow*, and *Redis*. |
| **A3.4 Deployment of the final solution in the EOSC Marketplace** | Thanks to placing project source code in repository system based on GIT, it will be easier to deploy the whole project at EOSC marketplace as software type under Apache license 2.0. |
| **A3.5 Smooth transfer** | The whole code was shared using the Git repository `GitHub`, and it allowed for smooth transfer to the EOSC marketplace. |
| **A3.6 Service ticket system** | Dedicated customer support system (https://glpi-project.org/ ticket system) is available for managing user tickets related to the web application. The ticket system is currently available only for internal purposes. |

## 3.2 Organization of work and resource

Table 6: Organization of work and resource

| **Resource** | **Roles and Past Responsibilities** |
|---|---|
| **Product Owner/Scrum Master** | Was responsible for defining and prioritizing the product backlog, ensured the team understood the requirements, and managed the agile/scrum processes. Worked as a liaison between stakeholders and the development team. |
| **Senior Backend Developer** | Was responsible for server-side application logic and integration of the front-end work. Designed and implemented APIs, managed databases, and ensured performance and reliability of backend infrastructure. |

Continued on the next page

Table 6 Organization of work and resource –Continued on the next page

| Resource | Roles and Past Responsibilities |
|---|---|
| **Senior Frontend Developer** | Managed the user interface and user experience of the application. Translated design mockups into interactive web pages, ensured optimal performance on various devices, and integrated with backend services. |
| **Senior IoT Developer** | Specialized in developing software for connected devices. Understood hardware-software integration, managed device data streams, and ensured reliable communication between devices and backend servers. |
| **Artificial Intelligence Expert** | Focused on designing, training, and implementing machine learning models. Utilized big data for analysis, predicted trends, and created intelligent applications that enhanced user experience. |
| **Tester** | Ensured the quality and reliability of the software through rigorous testing. Identified bugs, reported issues to the development team, and validated that solutions met the initial requirements. |

## 3.3 Quality control measures

Table 7: Quality control measures

| Concept | Availability and offered | Description to be carried out |
|---|---|---|
| **Implementation of a quality system** | Yes | We had used scrum agile technology for the project management. The team was relatively small (6 people), and this had been the proven methodology for short projects without large teams. To manage the planning, progress, status, and risk of work, we had used the Jira platform. To document the entire project, we had used the Confluence platform, which could be integrated with Jira.<br>Moreover, we had ensured three performance indicators:<br>1. Transmission and recording of images from type 1 and type 2 cameras, which worked in two ecosystems for a minimum of 1 month with a minimum frequency of 7 images per day, had less than a 2% error rate.<br>2. Pheno and AI result indicators from type 1 and type 2 cameras working in two ecosystems for a minimum of 1 month had no fewer than 3 pheno results and no fewer than 2 AI results.<br>3. The Service Level Agreement (SLA level) was maintained at the 99.5% level. This meant that periods of allowed downtime or unavailability were no more than 3h 37m 21s monthly.<br>For testing scenarios and test runs, we had used the TestRail platform, which could be integrated with Jira. Jira was used to generate team project timesheets for each project member. The reporting had been completed by the following Monday EOD every two weeks using Jira (with two-week-long sprints). Planning had been conducted every two weeks before the sprint started. We had implemented the ISO9001:2015 standard and followed the rules in this quality management certificate for software development. The tenderer possessed Dekra's certificate confirming this. |

Continued on the next page

Table 7 Quality control measures –Continued on the next page

| **Concept** | **Availability and offered Description** | |
|---|---|---|
| **Risk management and continuity of the service in case of absence of the member of the team dedicated to a particular task** | Yes | A team of 6 people had been selected to develop the project within this tender. Nevertheless, the tenderer had more developers (at least 3 more) who could substitute the developers dedicated to the project, and more testers (at least 2 more) who could replace the tester assigned to this project. In a similar vein, some of the team members could assume alternative roles if necessary. The Senior IoT Developer could act as a product owner, and one of the developers could assume the AI Expert role. Therefore, the arrangement of the team was flexible, in accordance with the rules of the scrum agile methodologies. |
| **Measures to ensure the performance and maintenance of the software for at least 12 months after the end of the contract.** | Yes | The platform was maintained on the servers owned by us. The tenderer had a fully redundant server room with sufficient storage and computational power. It was properly secured using Fortigate solutions. We have two independent optical fibers from two different internet providers, with an LTE backup line for situations where fiber channels internet ceased. They had a three-stage power supply. The tenderer provided 24x7x365 services from this server room. |
| **Measures to ensure compliance with the data protection regulation.** | Yes | The Product Owner has the leading auditor for ISO27001:2017 certificate from Dekra. This meant that we knew (and used) the rules of information security management systems. |

# 4 Contact

If you need to get in touch with our team, please use the appropriate email address below based on the nature of your inquiry:

- **Administrative Inquiries:** For questions related to licensing, partnerships, and other administrative matters, please contact ai4pheno-admin@seth.software.
- **General Information:** For general questions or information about our software and its features, reach out to ai4pheno-info@seth.software.
- **Security Issues:** If you have identified a security vulnerability or have concerns about the security of our software, please alert our security team immediately at ai4pheno-security@seth.software.
- **Support:** For technical support, troubleshooting, or reporting bugs, get in touch with our support team at ai4pheno-support@seth.software.

We aim to respond to all inquiries in a timely manner. Thank you for your interest in AI4Pheno.