APPROVAL

Role	Name	Function	Signature
Author	Evmorfia Kilimtzidi	QA	
Review	Lara Piers	R&D PM	
Review	Julien Finci	CTO	
Approval	Mauro Rinaldi	RAQM	

PURPOSE

The purpose of this procedure is to describe and define responsibilities, requirements, methods, and phases in software development in ASPIVIX SA. The goal is to ensure that the software is safe, effective, and in compliance with appropriate requirements, regulations, and standards.

SCOPE

This procedure covers the activities involved in the development, management, modification and documentation of medical device software.

RESPONSIBILITIES

Responsible for implementing and maintaining this SOP is the head of R&D (CTO). All parties participating must respect the defined procedure.

DOCUMENT HISTORY

Description of Changes	Version
Initial version	A

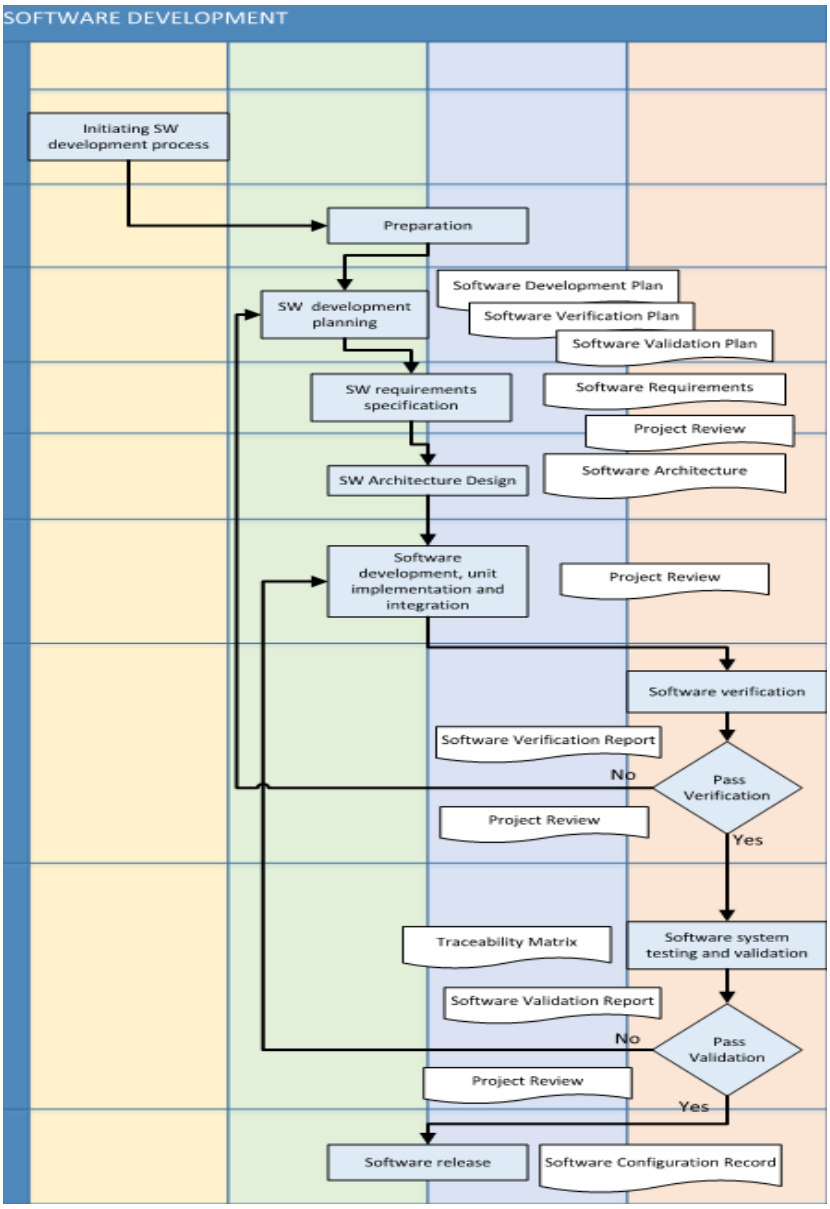
TABLE OF CONTENT

1. PROCEDURE

The process conforms to IEC 62304 and contains all activities, tasks, and evidence foreseen for medical device software designed and developed by ASPIVIX SA.

ASPIVIX SA needs to ensure the development and maintenance of medical software during the lifecycle according to this procedure and INS-208-1 Software Maintenance.

Figure 1: Software Development flowchart



The procedure starts when the initiative for the development of either brand-new software or new software features of existing software products are given. Such initiative or request can be placed:

- as a consequence of a request from a customer or goal to place a new software product on the market.
- as a consequence of market analysis identifying a new requirement.
- as a consequence of clinical data collection.
- as a consequence of a claim that software is not performing as defined.

Software development initiatives can be related either to new development projects or requests for changing existing software. New projects must be accepted and approved by the CEO. Request for change must be approved by the Project Manager before development activities start.

Table 1 – Stage Summary

Phase	Activities	Outputs	Milestones
Planning	Product Definition Identification of main product requirements	Initial Project requirements Software Development Plan Software Verification Plan	Project Approval
Requirements	Identification of software requirements Preliminary hazard analysis	Software Requirements Spec Updated Risk Analysis Primary Traceability Analysis	SRS approval
Design	Definition of the system architecture (top-level) Updated hazard analysis Test definition	Software Architecture Test Plan-System level (updated verification plan) Updated Risk Analysis Updated SRS Updated Traceability Analysis	Project Review
Implementation	Detailed design (comments in source code) Coding Coding Verification Test plan – Unit (module)	Program Code (Executable + source code) Software testing report (module Level verification/updated software verification plan)	Project Review
Verification and Validation	Integration Level testing System Level testing	Software Testing Report-Integration Level Verification Software Testing Report - System level Validation (software verification plan) Anomaly List Updated Software Requirements Updated Software Architecture Updated Risk Analysis Traceability Matrix	Approval Release meeting
Software release	Design freeze	Final Software Release/Project Review	Software release

1.1. Software Development Planning

Class	Definition
Class A	– the software system cannot contribute to a hazardous situation; or – the software system can contribute to a hazardous situation which does not result in unacceptable risk after consideration of risk control measures external to the software system.
Class B	– the software system can contribute to a hazardous situation which results in unacceptable risk after consideration of risk control measures external to the software system and the resulting possible harm is non-serious injury.
Class C	– the software system can contribute to a hazardous situation which results in unacceptable risk after consideration of risk control measures external to the software system and the resulting possible harm is death or serious injury.

1.1.1. Software safety classification

Classification of the software according to IEC 62304:2006/A1:2015 Section 4.3 shall be determined to be able to understand the requirements for the documentation and planning required.

Software shall be defined as either:

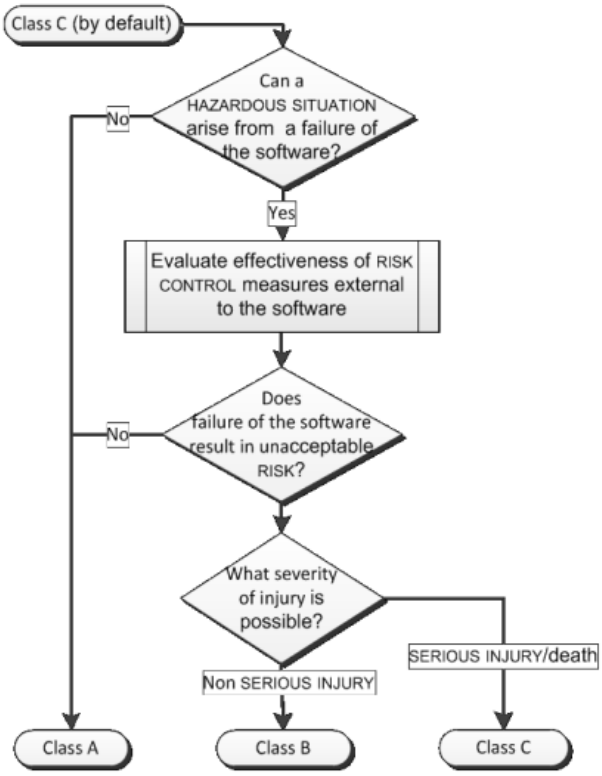


Figure 2: Software safety classification (according to IEC 62304)

The classification of medical software needs to be documented in the software development plan and affects the level of process documentation that will be required in preparation for release submission. Software products classified as Class B will also have their own separate Software Risk Analysis that will be linked to the overall Product Risk Analysis.

Once this has been determined, the following table shall be used to define the minimum requirements for the classification of software.

Note: these are the minimum requirements for the classification of software and contractual or other regulatory requirements may need additional documentation.

Software Documentation	Class A	Class B	Class C
Software development plan	Must contain contents of sections 5.1 IEC 62304:2006/A1:2015. The plan's content list increases as the class increases, but a plan is required for all classes.		
Software requirements specification	Software requirements specification conforming to 5.2 IEC 62304:2006/A1:2015. The content list for the software requirements specification increases as the class increases, but a document is required for all classes.		
Software architecture	Not required	A software architecture to 5.3, refined to software unit level for Class C	
Software	Not required	Refine software	Document detailed

detailed design		architecture to software units	design for software units
Software unit implementation	All units are implemented, documented, and source-controlled	Additional software unit acceptance	
Software unit verification	Not required	Define process, tests, and acceptance criteria (5.5.2, 5.5.3). Carry out verification	Define additional tests and acceptance criteria (5.5.2, 5.5.3, 5.4.4). Carry out verification
Software integration and integration testing	Not required	Integration testing to 5.6	
Software system testing	System testing to 5.7		
Software release	Document the version of the software product that is being released	List the remaining software anomalies, annotated with an explanation of the impact on safety or effectiveness, including operator usage and human factors	

According to the FDA, recommended documentation for a premarket submission depends on the device's risk to a patient, a user of a device, or others in the environment of use. FDA intends to take a risk-based approach to help determine the device's Documentation Level, which is either **Basic** or **Enhanced.**

Enhanced Documentation should be provided for any premarket submission that includes device software function(s) where a failure or flaw of any device software function(s) could present a hazardous situation with a probable risk of death or serious injury, either to a patient, user of the device, or others in the environment of use. These risks should be assessed prior to the implementation of risk control measures. Risks should be considered in the context of the device's intended use (e.g., impacts to safety, treatment, and/or diagnosis), and other relevant considerations.

Basic Documentation should be provided for any premarket submission that includes device software function(s) where Enhanced Documentation does not apply.

Software Documentation Elements	Basic Documentation Level	Enhanced Documentation Level
Documentation Level Evaluation	A statement indicating the Documentation Level and a description of the rationale for that level.	
Software Description	Software description, including an overview of significant software features, functions, analyses, inputs, outputs, and hardware platforms.	
Risk Management File	Risk management plan, risk assessment demonstrating that risks have been appropriately mitigated, and risk management report.	

Software Requirements Specification (SRS)	SRS documentation, describing the needs or expectations for a system or software, presented in an organized format, at the software system level or subsystem level, as appropriate, and with sufficient information to understand the traceability of the information with respect to the other software documentation elements (e.g., risk management file, software design specification, system and software architecture design chart, software testing).	
System and Software Architecture Design	Detailed diagrams of the modules, layers, and interfaces that comprise the device, their relationships, the data inputs/outputs and flow of data, and how users or external products (including information technology (IT) infrastructure and peripherals) interact with the system and software.	
Software Design Specification (SDS)	FDA is not recommending the SDS as part of the premarket submission. The sponsor should document this information on the design via the DHF for the device. During premarket review, the FDA may request additional information, if needed, to evaluate the safety and effectiveness of the device.	SDS documentation, including sufficient information that would allow FDA to understand the technical design details of how the software functions, how the software design completely and correctly implements all the requirements of the SRS, and how the software design traces to the SRS in terms of intended use, functionality, safety, and effectiveness.
Software Development, Configuration Management, and Maintenance Practices	A summary of the life cycle development plan and a summary of configuration management and maintenance activities; **OR** A Declaration of Conformity to the FDA-recognized version of IEC 62304, including subclauses 5.1.1 - 5.1.3, 5.1.6 - 5.1.9, clause 6 (Software maintenance process), and clause 8 (Software configuration management process), among others as applicable.	**Basic Documentation Level, PLUS** complete configuration management and maintenance plan document(s); **OR** A Declaration of Conformity to the FDA-recognized version of IEC 62304, including subclause 5.1 (Software development planning), clause 6 (software maintenance process), and clause 8 (software configuration management process), among others as applicable.
Software Testing as Part of Verification and Validation	A summary description of the testing activities at the unit, integration, and system levels; **AND** System level test protocol including expected results, observed results, pass/fail determination, and system level test report.	**Basic Documentation Level**, **PLUS** unit and integration level test protocols including expected results, observed results, pass/fail determination, and unit and integration level test reports.
Software Version History	A history of tested software versions including the date, version number, and a brief description of all changes relative to the previously tested software version.	
Unresolved Software Anomalies	List of remaining unresolved software anomalies with an evaluation of the impact of each unresolved software anomaly on the device's safety and effectiveness.	

Table: Outline of Recommended Documentation (FDA)[1]

1.1.2. Software Development Plan

The Software Development Plan (5.1.1)[2], as part of Design Development Plan (T-200-1), shall be created and maintained (5.1.2) that establishes the controls within the project for the following:

- Processes to be used in the development of the software,
- Responsibilities and authorities,
- Deliverables, including the documentation of the activities and tasks (design and development stages),
- Traceability between any other system requirements, software requirements, system test, and any identified risk control measures,
- Configuration and change management inc. Software Of Unknown Provenance (SOUP) configuration items and software used in the development,
- Problem resolution activities for handling problems detected,
- The documentation required throughout the development shall be defined including, title, name, purpose, and procedures and responsibilities for development, review, approval and modification (5.1.8),
- Software configuration activities for the project based on the use of GitHub or similar tools (5.1.9 & 5.1.11),
- Supporting software configuration control (5.1.10) such as compiler/assembler versions, batch files, and environment settings.
- Resources needed.

The plan for software product development shall employ software engineering best practices in verification and validation, configuration management, reviews, project tracking, and software quality assurance. The project plan is available to all team members. The PM will use regular project meetings to maintain the status of the software development plan and to resolve any conflicts or changes that might occur.

The verification and validation activities shall be planned to include software integration activities and testing (5.1.5) and this shall be documented in T-200-6 Verification_Validation_Plan (5.1.6).

The software validation activities may include user testing (installation, operational and performance testing), project reviews, and customer sign-off.

The Design Development Plan (T-200-1) must define and shortly describe the utilized software development mode and include a reference to the software risk management requirements and activities (5.1.7).

Additionally, the PM will use review meetings to maintain the status of the software development plan and to resolve any conflicts or changes that might occur.

Project reviews shall be performed and documented by utilizing the T-200-4 Design Review_Minutes.

[1] Content of Premarket Submissions for Device Software Functions, Guidance for Industry and Food and Drug Administration Staff (https://www.fda.gov/media/153781/download)

[2] This procedure contains references in brackets (i.e. (5.2.1)) which are a direct reference to the requirements of EN 62304:2005/A1:2016

1.2.Software Requirements Specification

Software requirements specification is critical for the success or failure of software development projects. The requirements should be documented, actionable, measurable, testable, traceable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design and development.

If regulatory approval in some countries requires conformance to specific regulations or international standards, this conformance requirement should be documented in the software requirements.

Software Requirements Specifications (5.2) shall be developed as part of the template T-200-12 Design_Input_Requirements. This shall include (5.2.1):

- Functional and capability requirements i.e. performance, purpose, timings, code language, platform, OS, computing environment, compatibility with upgrades or SOUP
- Software system inputs and outputs i.e. data types, ranges, limits, defaults, previous similar design
- The interface between software system and other systems
- Software-driven alarms, warnings, or operator messages
- Security requirements i.e. authentication, authorization, audit trail, integrity
- Usability engineering requirements
- Data definition and database requirements
- Installation and acceptance criteria
- Methods of operations and maintenance
- User documentation to be developed
- User maintenance requirements
- Regulatory requirements.

The risk management analysis shall be reviewed and updated (5.2.3) and the risk control measures that have been implemented in software for potential defects documented (5.2.4).

The system requirements shall be reviewed and updated where necessary to ensure the requirements are fully documented (5.2.5).

It shall be verified and documented that the software requirements:

- Implement system requirements including those relating to risk control (not relevant for standalone software),
- Do not contradict one another,
- Are expressed in terms that avoid ambiguity,
- Are stated in terms that permit the establishment of test criteria and performance of tests to determine whether the test criteria have been met,
- Can be uniquely identified,
- Are traceable to system requirements or other sources.

1.3.Software Architecture Design

The Software design phase comprises, but not limited to:

- Global software architecture diagrams and schemes
- Physical architecture
- Logical architecture.

The Software Architecture Specification shall be created to meet software requirements and is part of the template Design Input Requirements_ T-200-12, define software structure, and identify software items.

This shall include:

- The interfaces between the software items and the components external to the software items (both software and hardware) (5.3.2)
- Detail on the functional and performance requirements of SOUP (5.3.3)
- Hardware and software required for the proper operation of SOUP i.e. processor type/speed, memory type/speed, communication, and display requirements (5.3.4)
- Any segregation required for risk management processes i.e. operations to work on different processors (5.3.5)
- Verify that the software architecture meets the software requirements (5.3.6)

The output of this activity is T-208-1 Software Architecture containing architecture description and justification for choosing the described architecture.

T-208-1 Software Architecture also contains a list of all SOUP items required for the development of software.

The PM and Development Team are responsible for software architecture.

1.4.Software Implementation

The software shall then be developed, following the plans and specifications developed above (5.5.1).

Strategies, methodologies, and procedures for verifying each software unit shall be established, verified, and documented (5.5.2) which shall include acceptance criteria (5.5.3). The software shall be verified to ensure that the software meets the acceptance criteria.

Software unit verification shall be undertaken and documented following planned verification activities (5.5.5).

1.5.Software Integration and Integration Testing

Three terms identify the software decomposition. The top level is the software system. The lowest level that is not further decomposed is the software unit. All levels of composition, including the top and bottom levels, can be called software items. A software system, then, is composed of one or more software items, and each software item is composed of one or more software units or decomposable software items.

PM is responsible for choosing which software feature-unit-item from the requirements list will be developed by which software developer. PM assigns the tasks to the software developer(s). Tasks contain sufficient descriptions of software units that should be developed or reference to the source where complete information can be found.

Each software item shall be tested prior to integration.

The software shall be integrated which shall be following the Integration Plan (5.6.1).

Note: Software integration and software system testing may be undertaken as a single plan (T-200-6 Verification_Validation_Plan) and set of activities and may occur during the implementation phase as long as the requirements below are met.

- The integration of the software shall then be verified to ensure that the software units have been integrated into the software items and software system and that the hardware items and support for manual operations have been integrated (5.6.2). This verification shall be documented and is likely to be in the form of a review/inspection of the code.
- The software items shall be tested in accordance with the integration plan, ensuring that the software performs as intended (5.6.4) and the results documented (5.6.3). The integration test procedure shall be evaluated for correctness to ensure the results have been achieved as expected (5.6.5).

- Where required, regression testing shall be undertaken and documented to demonstrate that defects have not been introduced on previously tested and verified software (5.6.6).
- Records of integration shall be maintained, including the test results, test criteria to enable it to be repeated i.e. test case specification showing actions and expected results, equipment and test environment, and identification of the tester (5.6.7).

Any anomalies found during software integration shall be documented and subject to a problem resolution process (5.6.8), as defined in the procedure INS-208-3 Software Problem Resolution Process.

1.6. Software System Testing

The software system shall be subject to a formal set of tests, including the definition of input stimuli, expected outcomes, pass/fail criteria, and results to ensure that all software requirements are covered (5.7.1).

Any anomalies found during software integration shall be documented and be subject to a problem-resolution process (5.7.2).

Changes made to the software as a result of the software system testing shall be evaluated and tests repeated (either the same or modified tests) to verify the effectiveness of the change; performance of additional tests to ensure no side-effects have been introduced and review the risk management activities (5.7.3).

The software system testing shall be realized in accordance with T-200-6 Verification_Validation_Plan and verified to ensure that the strategies have been appropriate, the test procedures trace to software requirements, all software requirements have been tested and all tests have met the pass/fail criteria (5.7.4).

Software verification can be initiated at a project meeting. PM is responsible for initiating software verification as soon as the software is ready to be verified. The request for verification shall contain information about software items that have to be verified (either the whole software system or specific software items).

The output of this activity is T-200-7_Verification_Validation_Summary_Report.

Records of integration shall be maintained, including the test results, test criteria to enable it to be repeated i.e. test case specification showing actions and expected results, equipment and test environment, and identification of the tester (5.7.5).

The validation of software shall be undertaken to ensure that the developed software meets the user requirements i.e. to determine if the right software product has been developed. This may be through user testing, project reviews, and customer acceptance and shall be documented in formative or summative study. Validation Planning activities are included in T-200-7_Verification_Validation_Summary_Report.

The software validation team performs functional evaluations and performance evaluations in a real implementation environment. The results of medical software validation activities shall be documented. Testing and validation results are reported in the T-200-7_Verification_Validation_Summary_Report.

Template T-200-10 Traceability Matrix is used to capture test procedure steps and test results as well as links to applicable requirements and risks.

1.7. Software Release

Prior to formal software release, the following shall be checked:

- Ensure that all verification activities have been completed and results reviewed (5.8.1)
- Document, evaluate, and risk assess any and all known anomalies (5.8.2; 5.8.3)

- Document the released versions (5.8.4) and how the released software was created (5.8.5)
- Ensure that all activities are complete along with all associated documentation (5.8.7)

For production release, the method of ensuring that the software can be delivered to the point of use without corruption or change shall be established, documented, and verified, including replication, media labeling, packaging, protection, storage, and delivery (5.8.8).

T-208-3 Software Configuration Record is prepared as documented output from this activity according to INS-208-2 Software Configuration Management.

1.8. Changes

Changes to software during the development process may come from additional requirements identified during design reviews, during development, or from customers. Changes are managed by SOP-303 Change Management.

1.9. Problem Analysis and Resolving

After reporting the problem with the medical software, support starts with problem analysis. If the problem is solved at this level the process is completed. If the problem is not solved the customer complaint handling process starts.

If the update of medical software is needed for problem solving, INS-208-3 Software Problem Resolution Process will be used.

After problem solving, ASPIVIX SA is responsible for informing the user about the status of the issue and addressing and closing the issue.

All records regarding technical support need to be maintained.

1.10. Design and development files

ASPIVIX SA shall keep and maintain all files developed during development activities with appropriate references that demonstrate conformity with relevant requirements, as well as records for design and development changes.

This file contains all evidence of a medical software version: software codes, items, files, protocols, documents, instructions, procedures for use and maintenance, etc. DMR also identifies all specifications and full descriptions of how to develop medical software, testing software, verification, validation, evaluation and label, and supporting activities.

2. REFERENCE

2.1. Procedures, instructions, and guidelines

- SOP-303 Change Management
- INS-208-1 Software Maintenance
- INS-208-2 Software Configuration Management
- INS-208-3 Software Problem Resolution Process

2.2. Templates and forms

- T-200-1 Design Development Plan
- T-200-4 Design Review_Minutes
- T-200-6 Verification_Validation_Plan
- T-200-7_Verification_Validation_Summary_Report
- T-200-10 Traceability Matrix
- T-200-12 Design_Input_Requirements
- T-208-1 Software Architecture
- T-208-3 Software Configuration Record

2.3.External

- IEC 62304:2006/A1:2015 - Medical device software – Software life cycle processes
- ISO 13485:2016 - Medical devices - Quality management systems - Requirements for regulatory purposes
- EN ISO 13485:2016/A11:2021 - Medical devices - Quality management systems - Requirements for regulatory purposes
- FDA QSR 21 CFR Part 820, Medical Devices - Quality System Regulation
- FDA Guidance" Content of Premarket Submissions for Device Software Functions"
- EN ISO 14971:2019/A11:2021 - Medical devices - Application of risk management to medical devices

Aspivix SA | Route de la Corniche 8 | Bureau 10 | 1066 Epalinges | Switzerland

Certificate of Completion

SOP-208 Software Development_rev.A.pdf

Printed on 2025-03-27 - 18:56:49 UTC

Document Details

Created by: Eva Kilimtzidi
Created on: 03/26/2025 12:04:38 (UTC)
Creation IP Address: 185.10.224.50
Signers: 4

Document ID: 495
Signature: 3ab47326e794d5e12cdcc21560afb48c9352e2b2c0479ecae7b5834e4abeeb2b

Participants

Signatory	Email	Email Verification
Eva Kilimtzidi	eva.kilimtzidi@aspivix.com	☑
	Signatory's hash: 5c82ba48970ca2c8f4cdea81ab2f5fe95d4c71c4d1f26c05a5d383489c3cb064	
Lara Piers	lara.piers@aspivix.com	☑
	Signatory's hash: 4cf15c2b40458f1c7e73e9a39a066eba60b53fe83bd792da24cd281a64489537	
Mauro Rinaldi	mauro.rinaldi@aspivix.com	☑
	Signatory's hash: 6b9921bea42aa332590bce216bd2c0b312f057d3f3d842f63729ba00ea23a8f8	
Julien FINCI	julien.finci@aspivix.com	☑
	Signatory's hash: 79a0931948a72f84aa0ac74abf56ed20b30f3369a15ceb8c5e8374e9158130ba	

Email Verification: The signatory has confirmed control of their email inbox by clicking on a unique link

Signing Events

Action	By	Date (UTC)	Geolocation	IP
Creation	Aspivix SA, Eva Kilimtzidi eva.kilimtzidi@aspivix.com	03/26/2025 12:04:38	46.3552000, 7.0121000 View	185.10.224.50
			Signature: 3ab47326e794d5e12cdcc21560afb48c9352e2b2c0479ecae7b5834e4abeeb2b	
Signature 	Aspivix SA, Eva Kilimtzidi eva.kilimtzidi@aspivix.com	03/26/2025 12:04:46	46.3552000, 7.0121000 View	185.10.224.50
			Signature: 8d2fd83025fb1094bda5fbc8e85f0a98c79276ad4c9e3b4cfd8c4189ea093e73	
Signature 	Aspivix SA, Lara Piers lara.piers@aspivix.com	03/26/2025 12:23:57	46.3552000, 7.0121000 View	185.10.224.50
			Signature: 79ab553099118dc1d7a5bd07d8cc5cc30036ec3d488e5f1e999c5fdd6c948d4e	
Signature 	Aspivix SA, Julien FINCI julien.finci@aspivix.com	03/26/2025 17:40:01	46.3552000, 7.0121000 View	185.10.224.50
			Signature: 2293b68506670751c90113a2f9bf26c5d02563afe3fb74c13a638e6fa5d41136	
Signature 	Aspivix SA, Mauro Rinaldi mauro.rinaldi@aspivix.com	03/27/2025 16:47:05	47.3643000, 8.5437000 View	172.226.132.54
			Signature: 82cf8892bc23fce7f63e8ccfbb84cce4ba65a1369ab8f1508d5bd6c29868dc1e	

✔ The document's integrity is valid.

The final document and this completion history have been sent by email on 03/27/2025 to: lara.piers@aspivix.com, eva.kilimtzidi@aspivix.com, julien.finci@aspivix.com, mauro.rinaldi@aspivix.com.

Access Logs

Viewed/downloaded by	Date (UTC)	State	Geolocation	ip
Aspivix SA, Eva Kilimtzidi eva.kilimtzidi@aspivix.com	03/26/2025 12:04:38	Before Signature	46.3552000, 7.0121000 View	185.10.224.50
Aspivix SA, Eva Kilimtzidi eva.kilimtzidi@aspivix.com	03/26/2025 12:04:42	Before Signature	46.3552000, 7.0121000 View	185.10.224.50
Aspivix SA, Lara Piers lara.piers@aspivix.com	03/26/2025 12:22:48	Before Signature	46.3552000, 7.0121000 View	185.10.224.50
Aspivix SA, Julien FINCI julien.finci@aspivix.com	03/26/2025 17:39:49	Before Signature	46.3552000, 7.0121000 View	185.10.224.50

Aspivix
Route de la Corniche 8
1066 Epalinges | Switzerland

contact@aspivix.com
https://aspivix.com

Tax ID CHE-347.247.288 TVA

Viewed/downloaded by	Date (UTC)	State	Geolocation	ip
Aspivix SA, Mauro Rinaldi mauro.rinaldi@aspivix.com	03/27/2025 16:46:58	Before Signature	47.3643000, 8.5437000 View	172.226.132.54
Aspivix SA, Eva Kilimtzidi eva.kilimtzidi@aspivix.com	03/27/2025 16:50:12	After Signature	47.3992000, 8.3956000 View	194.230.158.207
Aspivix SA, Eva Kilimtzidi eva.kilimtzidi@aspivix.com	03/27/2025 19:56:46	After Signature	46.4315000, 6.8825000 View	178.192.217.44

Aspivix
Route de la Corniche 8
1066 Epalinges | Switzerland

contact@aspivix.com
https://aspivix.com

Tax ID CHE-347.247.288 TVA