Ref. Ares(2026)3418646 - 31/03/2026

D7.3 – AccesS Acceptance Tests v1

This project has received funding from the European Union's Horizon Europe research and innovation programme under the grant agreement number 101147722. The European Union is not liable for any use that may be made of the information contained in this document, which is merely representing the authors' view.

Project Acronym:	AccesS
Project Full Title:	Enhancing Accessibility and Sustainability in Smart Cities and Smart Buildings: The Universal Accessibility Suite Initiative
Grant Agreement:	101147722
Project Duration:	36 months (01/06/2024 – 31/05/2027)

DELIVERABLE 7.3
AccesS Acceptance Tests v1

Work Package:	WP7 Advancing Accessibility and Sustainable Urban Environments
Task:	T7.3 Inclusive Toolbox -Testing
Document Status:	Final
File Name:	AccesS_D7.3_AccesS Acceptance Tests v1_IsZEB
Due Date:	31.03.2026
Submission Date:	31.03.2026
Lead Beneficiary:	IsZEB

Dissemination Level	
Public	☒
Sensitive	☐

Authors List

Leading Author(s)			
First Name	**Last Name**	**Beneficiary**	**Contact e-mail**
Afroditi	Zamanidou	IsZEB	a.zamanidou@iszeb.gr
Vasileios	Metavitsiadis	IsZEB	v.metavitsiadis@iszeb.gr

Co-Author(s)				
#	**First Name**	**Last Name**	**Beneficiary**	**Contact e-mail**
1	Panagiota	Chatzipanagiotidou	CERTH	phatzip@iti.gr
2	Napoleon	Bezas	CERTH	bezas@iti.gr
3	Tzilopoulos	Dimitris	CERTH	dimtzilop@iti.gr
4	Ioannis	Koskinas	CERTH	jkosk@iti.gr
5	Ilias	Aliferis	UNIS	aliferisi@unisystems.gr
6	Dimitris	Botsis	UNIS	botsisd@unisystems.gr

Reviewers List

Reviewers			
First Name	**Last Name**	**Beneficiary**	**Contact e-mail**
Emmanouil	Zidianakis	FORTH	zidian@ics.forth.gr
Guilherme	Munhoz	DMO	guilherme@demobv.nl
Alejandra Losada	Jiménez	CETEM	a.losada@cetem.es
Laura Casolo	Ginelli	HIVE	laura.casologinelli@hivepower.tech

Version History

v	Author	Date	Brief Description
0.1	Afroditi Zamanidou, Vasileios Metavitsiadis (IsZEB)	15.12.2025	Table of Contents
0.2	Afroditi Zamanidou, Vasileios Metavitsiadis (IsZEB)	19.02.2026	Initial Draft

0.3	All authors	13.03.2026	Consolidated Draft with input in all sections
0.5	Vasileios Metavitsiadis (IsZEB); Ioannis Koskinas (CERTH)	23.03.2026	Draft for internal review
0.6	Guilherme Munhoz (DMO); Emmanouil Zidianakis (FORTH)	30.03.2026	Internally reviewed version
0.7	Afroditi Zamanidou, Vasileios Metavitsiadis (IsZEB)	31.03.2026	Updated based on final comments from reviewers
1.0	Vasileios Metavitsiadis (IsZEB); Panagiota Chatzipanagiotidou, Dimos Ioannidis (CERTH)	31.03.2026	Final version for submission

Legal Disclaimer

The AccesS project has received funding from the European Union's Horizon Innovation Action under grant agreement No 101147722. The sole responsibility for the content of this publication lies with the authors. It does not necessarily reflect the opinion of the European Climate, Infrastructure and Environment Executive Agency (CINEA) or the European Commission (EC). CINEA or the EC are not responsible for any use that may be made of the information contained therein.

Copyright

© IsZEB, 6th km Harilaou-Thermis, 57001, Thessaloniki, Greece. Copies of this publication – also of extracts thereof – may only be made with reference to the publisher.

Executive Summary

Deliverable 7.3 “AccesS Acceptance Tests v1” presents the first version of the acceptance testing activities of the AccesS Suite, carried out as part of Task 7.3 “Inclusive Toolbox – Testing”. The objective of this deliverable is to document the testing methodology, execution, and results of unit, integration, security, and system workflow tests performed in the CERTH/ITI nZEB Smart House, a facility equipped with advanced IoT technologies.

The testing process followed a structured Acceptance Test Plan, covering the AccesS Suite and its components. Unit tests verified the correct operation of individual components, while integration tests ensured seamless communication between services, including authentication via Keycloak and data exchange through APIs. Security testing confirmed the robustness of authentication, authorization, and data protection mechanisms, and system testing validated end-to-end workflows across the platform.

The results demonstrate that the AccesS Suite operates reliably and consistently under the tested conditions. The testing also confirmed the effectiveness of the platform’s security mechanisms and the ability to support complete user workflows within a distributed system architecture. Core functionalities across components were successfully validated, demonstrating stable performance. All tools were fully integrated within the platform, while for the case of the large-scale Accessibility Assessment Tool, the tool progressed towards full integration and has been delivered in M22 and therefore, currently it has been validated at unit level, while additional testing is recommended in the next phase.

Findings indicate that the AccesS Suite has achieved the technical maturity required for subsequent validation phases. The second version of this deliverable will present the next steps, which will focus on completing the integration testing of all components, enhancing system robustness, and refining performance to support large-scale deployment.

Table of Contents

1 Introduction 11
1.1 Scope and objectives of the deliverable 11
1.2 Structure of the deliverable 11
1.3 Relation to Other Tasks and Deliverables 12
2 Methodology 13
2.1 Testing Overview 13
2.2 Test Strategy and Approach 15
2.3 Test Environment and Setup 16
2.3.1 Test Infrastructure 16
2.3.2 Template 18
3 Test Execution, Results and Analysis 19
3.1 AccesS Suite 19
3.1.1 Testing 19
3.1.2 Assessment 22
3.2 Energy Efficient & Sustainable Design Tool 22
3.2.1 Testing 23
3.2.2 Assessment 24
3.3 Energy Comfortness Tool 24
3.3.1 Testing 25
3.3.2 Assessment 27
3.4 ErgoFlow 27
3.4.1 Testing 28
3.4.2 Assessment 29
3.5 Virtual Humans (VHs) 29
3.5.1 Testing 29
3.5.2 Assessment 30
3.6 3D Simulation Environments 30
3.6.1 Testing 31
3.6.2 Assessment 31
3.7 Adaptive Building Operation & Management Systems 32
3.7.1 Testing 32
3.7.2 Assessment 33

3.8 EV Charging Optimizer 34
3.8.1 Testing 34
3.8.2 Assessment & Recommendations 35
3.9 Accessibility Assessment tool 35
3.9.1 Testing 36
3.9.2 Assessment 37
4 Conclusions 38
ANNEX A: Tests 39
AccesS Suite 39
Energy Efficient & Sustainable Design Tool 45
Energy Comfortness Tool 47
ErgoFlow 52
VUMs/VHs 54
3D Simulation Environments 58
BOMS TESTS 60
EV Charging Optimizer 67
Accessibility Assessment Tool (AAT) 70

List of Figures

Figure 1. Processing (top), Interoperability (middle), and Infrastructure/Physical (bottom) Layers of the AccesS Suite. 13
Figure 2. Representation Layer of the AccesS Suite. 14
Figure 3. Gantt Chart of the Acceptance Test Plan for the first phase of T7.3. 15
Figure 4. CERTH/ITI nZEB Smart House. 17
Figure 5. Screenshot from the AccesS Suite, showing the integrated tools and services. 19
Figure 6. Screenshot from EESDT, showing the project selection interface. 22
Figure 7. Screenshot from ECT simulation dashboard's starting screen. 25
Figure 8. Screenshot of ErgoFlow's Evaluation Tool. 27
Figure 9. Screenshot of ErgoFlow Library. 28

Figure 10. Screenshot from the VH creation within the Unit testing framework. 29

Figure 11. Screenshots from SANTO Tool. 31

Figure 12. Screenshot from BOMS. 32

Figure 13. Screenshot from the EV Charging Optimizer's user interface, depicting characteristics of different charging stations. 34

Figure 14. Screenshot from the EV Charging Optimizer's user interface, depicting hourly consumption. 35

Figure 15. Screenshots from the Accessibility Assessment tool's user interface. 36

List of Tables

Table 1. Test Description Template. 18

Table 2. AccesS Suite Unit Testing 39

Table 3. AccesS Suite Security Testing 40

Table 4. AccesS Suite Integration Testing 41

Table 5. AccesS Suite System Testing 43

Table 6. EESDT integration testing 45

Table 7. Energy Comfortness Tool Unit Test 47

Table 8. Energy Comfortness Tool Integration & Security Test 48

Table 9. Energy Comfortness Tool Integration Test 49

Table 10. Energy Comfortness Tool System Test 51

Table 11. ErgoFlow Ergonomics, Comfort and Accessibility Evaluation and Library Validation Test ... 52

Table 12. VUMs/VHs GUI Editor System and Integration Test Description 54

Table 13. VUMs/VHs GUI Editor Security Test Description 56

Table 14. 3D Simulation Environments Unit Test 58

Table 15. 3D Simulation Environments Integration Test 58

Table 16. BOMS Unit Testing 60

Table 17. BOMS Integration Testing 61

Table 18. BOMS Security Testing 62

Table 19. BOMS System Workflow Testing – End-to-End Operation 64

Table 20. Integration and functional testing of the EV Charging Optimizer 67

Table 21. AAT System and Integration Test 70

Table 22. AAT Security Test Description 72

List of Acronyms and Abbreviations

Term	Description
API	**Application Programming Interface**
ATP	**Acceptance Test Plan**
AAT	**Accessibility Assessment Tool**
BOMS	**Building Operation Management Systems**
DBL	**Data Base Layer**
DT	**Digital Twin**
E2E	**End-to-End**
ECT	**Energy Comfortness Tool**
EV	**Electric Vehicle**
GPU	**Graphics Processing Unit**
HTTP	**Hypertext Transfer Protocol**
HVAC	**Heating, Ventilation, and Air Conditioning**
HW	**Hardware**
IoT	**Internet of Things**
JSON	**JavaScript Object Notation**
JWKS	**JSON Web Key Set**
JWT	**JSON Web Token**
MQTT	**MQ Telemetry Transport**
nZEB	**nearly Zero Energy Building**
OS	**Operating System**
PDF	**Portable Document Format**
RBAC	**Role-Based Access Control**

REST	**Representational State Transfer**
SSO	**Single Sign-On**
SW	**Software**
URL	**Uniform Resource Locator**
VH	**Virtual Humans**

1 Introduction

1.1 Scope and objectives of the deliverable

Deliverable D7.3, “AccesS Acceptance Tests v1”, documents the scope, execution, and outcomes of the acceptance testing activities performed within Task T7.3 “Inclusive Toolbox – Testing”. The deliverable addresses the systematic verification and validation of the AccesS Suite components and services in a controlled environment, prior to their deployment in real-world settings.

Testing activities were mainly carried out at the CERTH/ITI nZEB Smart House at CERTH premises, while some unit tests were performed capitalizing AccesS pilot sites where applicable, such as the Droom je Thuis pilot case. The nZEB Smart House is representative of a single-family, detached residential building and is equipped with a wide range of IoT devices and smart home solutions, enabling realistic assessment of system interoperability, performance, security, and end-to-end workflows under operational conditions. This environment allows the evaluation of how the AccesS components and services interact and perform as an integrated system.

The scope of this deliverable includes unit, integration, security, and system workflow testing of the AccesS Suite, following an Acceptance Test Plan that defines the testing methodology, test cases, acceptance criteria, timing, and responsibilities. The testing process follows a structured sequence comprising: (i) unit tests of individual components, (ii) integration tests to verify seamless coordination and interoperability, (iii) security tests to identify and mitigate software vulnerabilities, (iv) system tests and technical assessment of end-to-end workflows, (v) formulation of recommendations for final refinements, and (vi) fine-tuning and overall optimization of the suite.

The overall objective of D7.3 is to ensure that the integrated AccesS Suite functions correctly, securely, and reliably as a whole, and that it meets its functional and technical requirements before actual deployment. Specifically, the deliverable aims to verify the correct operation of individual components, validate interoperability and data consistency across services, assess system-level behavior and workflows, and minimize security risks. In addition, it provides a technical assessment of testing outcomes and concrete recommendations for improvements, thereby reducing deployment risks and increasing confidence in the robustness, security, and readiness of the AccesS Suite.

1.2 Structure of the deliverable

1. **Introduction.** This chapter provides an overview of the deliverable's objectives, highlighting the relationships and dependencies between T7.3 (and D7.3) and other WPs or tasks within the project. It also outlines the structure of the deliverable, summarizing the key information included in each chapter.
2. **Methodology**. This chapter details the methodology used for the acceptance testing, including the test strategy and approach, the scope of testing, and the test environment, setup, and infrastructure.
3. **Test Execution, Results and Analysis**. This chapter presents the execution and findings of the unit, integration, security, and system tests. It provides descriptions of the tests conducted, the results obtained, and key outcomes. Also, it offers a comprehensive analysis of the test results, including comments or recommendations for improvements.
4. **Conclusions.** This chapter presents the conclusions drawn from the testing process, emphasizing the readiness and robustness of the AccesS Suite for deployment.

Annex. The annex section contains the filled test templates for each of the tested components.

1.3 Relation to Other Tasks and Deliverables

Deliverable D7.3 “AccesS Acceptance Tests v1” is closely related to several tasks across the AccesS project that deal with the development and integration of the AccesS Suite’s components, including the AccesS Suite itself. Since the acceptance tests presented in D7.3 verify the correct operation and interoperability of the AccesS Suite’s integrated components, this deliverable is mainly linked to T7.2 “AccesS Inclusive Toolbox – Integration of Components”, which deals with the integration of the different technologies developed in the project into the AccesS Suite. In addition, D7.3 evaluates tools and systems developed in other work packages, including T4.1 “Energy Efficient and Sustainable Design Tool”, T4.2 “Energy Comfortness Tool”, and T4.3 “Ergonomics and Comfort Design of Products”, from WP4; T5.1 “Virtual User Models and VUM-based Virtual Humans” and T5.2 “3D Simulation Environments” from WP5; and T6.1 “Adaptive Building Operation and Management System” and T6.3 “Integrated e-Mobility” from WP6 as well as T7.1 “Large-Scale Accessibility Assessment”. The tools developed in these tasks are integrated into the AccesS Suite and their acceptance tests are presented in this deliverable to ensure proper functionality and interoperability.

2 Methodology

2.1 Testing Overview

The primary objective of the Acceptance Tests is to provide a comprehensive evaluation of the AccesS services and components, ensuring seamless coordination and interoperability in a controlled environment prior to actual deployment. This evaluation is essential to verify the correct operation, performance, and security of the AccesS Suite.

The AccesS Suite is a centralized digital platform that combines various innovative tools to help design and evaluate buildings and urban spaces. Its primary goal is to ensure these environments are barrier-free, energy-efficient, and easy to navigate for everyone. Integrating multiple advanced technologies into a single platform involves several technical, organizational, and human-centered challenges. To guarantee that all the technologies are integrated and managed independently while still working together as a unified system, AccesS Suite follows a Layered Conceptual Architecture Perspective, shown in Figure 1 and Figure 2.

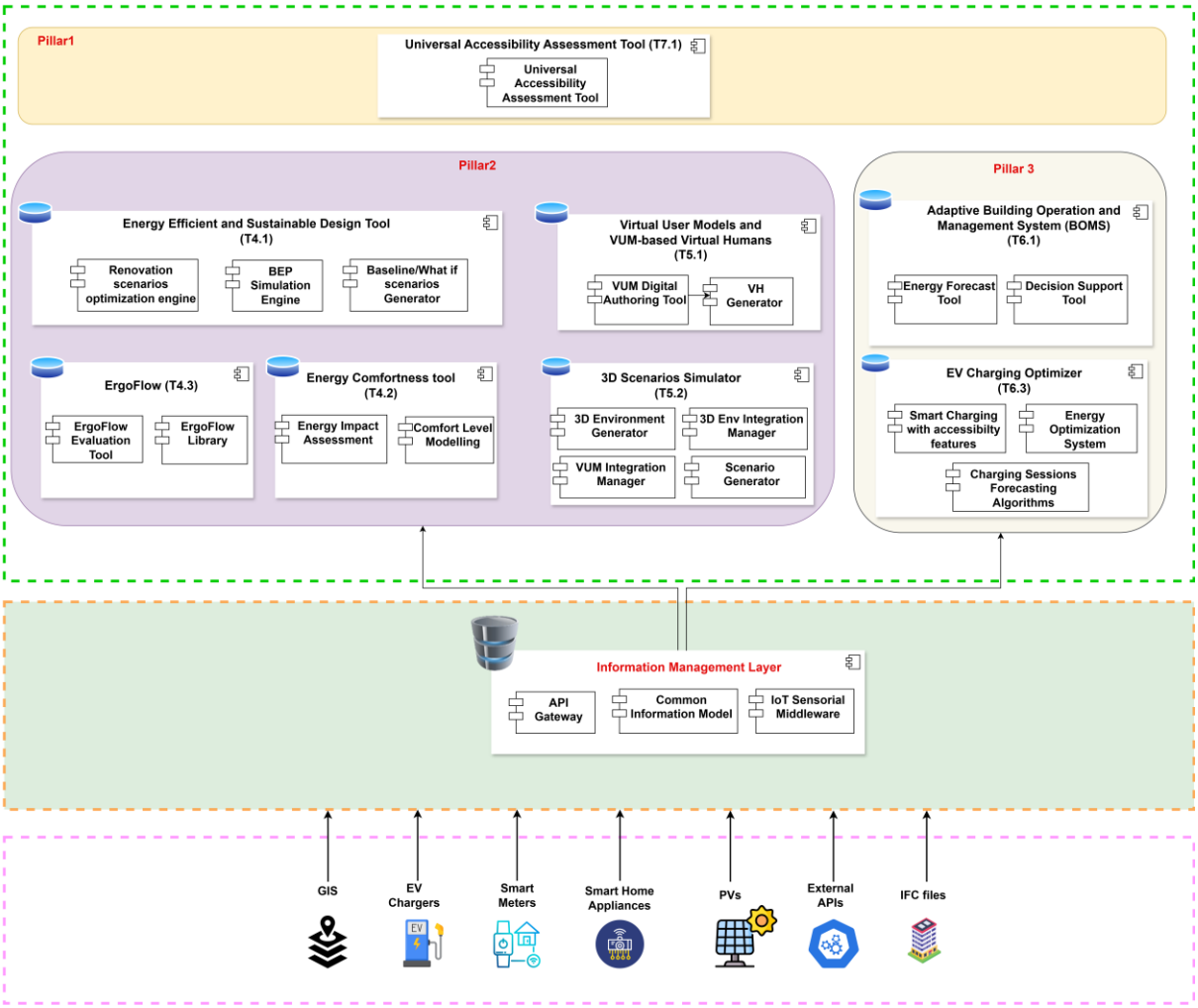


Figure 1. Processing (top), Interoperability (middle), and Infrastructure/Physical (bottom) Layers of the AccesS Suite.

The Layered Conceptual Architecture Perspective offers a structured overview of how the various tools and systems within the AccesS Suite interact and integrate. It illustrates how user interfaces, data

processing components, and underlying technologies collaborate to enhance accessibility, energy efficiency, and comfort in smart environments. This approach facilitates the development of simulations and AI-powered solutions, supporting more effective building management and the sustainable design of the built environment.

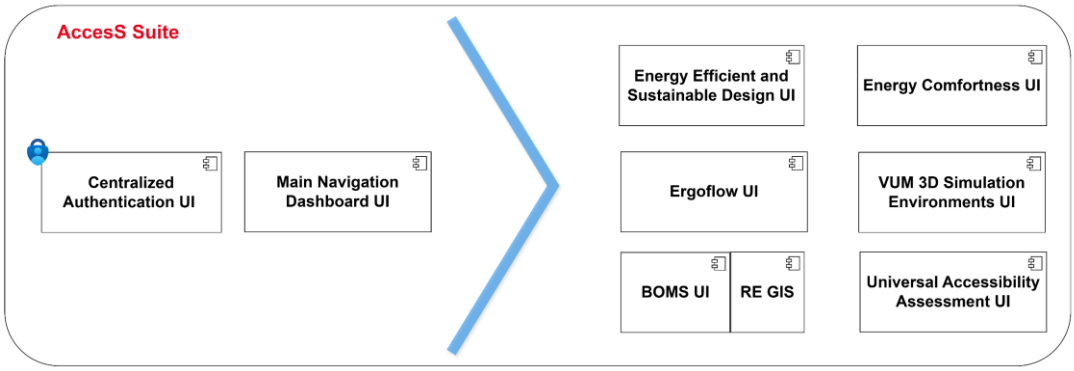


Figure 2. Representation Layer of the AccesS Suite.

The acceptance testing process comprises a set of detailed tests, including unit tests, integration tests, security tests, and system workflow tests. These tests are conducted at the CERTH/ITI nZEB Smart House, a state-of-the-art facility equipped with advanced IoT and smart home technologies representative of a typical residential building. This setting enables realistic testing conditions and supports the early identification and mitigation of issues that may arise during real-world operation.

An Acceptance Test Plan (ATP) governs the testing process, defining the testing methodology, scheduling activities, and establishing acceptance criteria to ensure a structured and systematic quality assurance approach. The main components of the testing process are:

1. **Unit Tests**: These tests validate individual components of the AccesS Suite in isolation, ensuring that each component functions according to its design specifications. Unit tests support early detection and resolution of issues before component integration.

2. **Integration Tests**: Integration tests assess interactions among integrated components, verifying correct data exchange and coordinated operation. These tests help identify issues related to interoperability, such as communication failures or data inconsistencies.

3. **Security Tests**: Security tests evaluate the security posture of the AccesS Suite by identifying vulnerabilities and weaknesses. The goal is to ensure protection of sensitive data and resilience against potential cyber threats through activities such as vulnerability scanning and validation of security mechanisms.

4. **System Tests**: System tests assess the overall functionality and performance of the integrated components within AccesS Suite and the AccesS Suite as a whole integrated system. These include end-to-end workflow validation, performance evaluation, and stress testing under realistic conditions.

The overall goal of the acceptance testing process is to identify and address potential issues, optimize system behavior, and provide recommendations for final refinements. This structured approach ensures that the AccesS Suite is robust, secure, and ready for deployment. By achieving these objectives, the project supports improved renovation efficiency, reduction of the gap between as-designed and as-built performance, and enhanced synergies across the construction and renovation value chain. The acceptance testing process provides confidence that the AccesS Suite meets the quality standards required for real-world application and supports the project's overall objectives.

2.2 Test Strategy and Approach

The acceptance test strategy and approach for the AccesS Suite are designed to ensure thorough evaluation of all components and services with respect to functionality, performance, interoperability, and security. The testing framework follows a structured and methodical approach, validating the system progressively from individual components to the fully integrated suite, thereby ensuring readiness for deployment.

The acceptance testing framework encompasses multiple test types—unit, integration, security, and system tests—each addressing specific objectives and system aspects:

- **Unit Tests**: Validate the functionality of individual components (e.g., the DBL), ensuring correct behavior in isolation.

- **Integration Tests**: Verify interactions between integrated components (e.g., communication between the Eclipse SensiNact Platform and the DBL), ensuring consistent data exchange and cohesive operation.

- **Security Tests**: Assess the security posture and identify vulnerabilities (e.g., authentication mechanisms in the DBL), safeguarding against cyber threats and ensuring data confidentiality and integrity.

- **System Tests**: Evaluate the overall functionality and performance of the AccesS Suite through end-to-end testing, ensuring compliance with user and system requirements under realistic operating conditions.

The ATP defines the overall testing strategy and approach, including a structured timeline (shown in Figure 3), to ensure comprehensive validation of all developed components and their interactions prior to deployment.

		M12	M13	M14	M15	M16	M17	M18	M19	M20	M21	M22
		May 2025	June 2025	July 2025	Aug 2025	Sep 2025	Oct 2025	Nov 2025	Dec 2025	Jan 2026	Feb 2026	March 2026
T7.3	Inclusive Toolbox –Testing											
ST7.3.1	Definition of the Acceptance Test Plan											
ST7.3.2	Unit tests											
ST7.3.3	Integration tests											
ST7.3.4	Security tests											
ST7.3.5	System tests and technical assessment of the outcome											
ST7.3.6	Recommendations for the final suite refinements											
ST7.3.7	Fine-tuning and overall optimization											
ST7.3.8	ToC for D7.3/Allocation of contributions											
ST7.3.9	First Draft of D7.3											
ST7.3.10	Review and Final Draft of D7.3											
ST7.3.11	Adress comments, finalize and submit D7.3											

Figure 3. Gantt Chart of the Acceptance Test Plan for the first phase of T7.3.

The testing methodology for the AccesS Suite emphasizes robustness and reliability. Unit tests focus on validating individual components, such as the DBL, deployed on remote servers at CERTH premises

using MongoDB. These tests include operations such as creating, retrieving, updating, and deleting building-related information using JSON-formatted data to reflect realistic usage scenarios.

Security testing is critical for assessing system resilience against vulnerabilities and unauthorized access. These tests include verification of encrypted password storage, authentication mechanisms, and token-based authorization within the DBL, ensuring that only trusted components can access system resources. System tests provide end-to-end evaluation of overall system functionality and performance, including continuous data exchange using the MQTT protocol.

The successful execution of the ATP for the AccesS Suite requires coordinated use of hardware, software, and personnel resources. Hardware resources include servers for hosting deployed AccesS components, while software resources comprise testing and automation tools for unit, API, integration, and system testing, as well as continuous integration and monitoring tools to support efficient execution, reporting, and real-time performance and security assessment. Personnel resources involve a multidisciplinary team, including test engineers responsible for test design and execution, software developers addressing issues identified during testing, security experts focusing on vulnerability assessment and mitigation, and project managers coordinating activities to ensure alignment with project timelines and objectives. This collaborative effort ensures that the AccesS Suite meets high standards of functionality, performance, interoperability, and security, supporting its successful deployment and operation.

Acceptance criteria for the AccesS Suite are defined to ensure compliance with functional, performance, security, and interoperability requirements. All components must operate according to their specified functionality, meeting defined performance benchmarks under expected load conditions. Security criteria require the absence of critical vulnerabilities and confirmation that appropriate protection mechanisms are in place. Interoperability criteria ensure seamless interaction and data exchange among all integrated components. Collectively, these criteria confirm that the AccesS Suite is robust, secure, and ready for deployment.

2.3 Test Environment and Setup

The test environment and setup for the AccesS Suite are designed to closely replicate real-world operating conditions and to support a thorough evaluation of system functionality, performance, interoperability, and security. By leveraging a controlled environment equipped with advanced IoT and smart home technologies, the testing process enables realistic scenario simulation and early identification of potential issues prior to deployment. This section describes the hardware and software environment, configurations, and tools used during the acceptance testing activities.

2.3.1 Test Infrastructure

The test infrastructure for the AccesS Suite integrates both hardware and software components to provide a stable and reliable environment for executing acceptance tests. The core of this infrastructure is the CERTH/ITI nZEB Smart House, a state-of-the-art facility that emulates a typical residential building and is equipped with advanced IoT and smart home solutions. The nZEB Smart House serves as a realistic testbed for evaluating the AccesS Suite under controlled yet representative conditions, enabling comprehensive testing that closely reflects real-world operation.

The nZEB Smart House incorporates a range of advanced building features designed to enhance energy efficiency and sustainability. These include high-performance insulation, energy-efficient windows, a solar photovoltaic system for on-site renewable electricity generation, and an advanced heating, ventilation, and air conditioning (HVAC) system. Together, these features ensure efficient building operation while maintaining optimal indoor environmental conditions, providing a relevant context for testing AccesS functionalities related to building performance and energy management.

Figure 4. CERTH/ITI nZEB Smart House.

Smart home technologies and data acquisition capabilities are central to the nZEB Smart House setup. The building integrates a variety of smart devices, such as thermostats, lighting systems, security components, and home automation solutions, all interconnected through a central IoT platform. This infrastructure enables seamless communication and coordinated control of devices. In addition, a dense network of sensors continuously monitors parameters including temperature, humidity, energy consumption, and indoor air quality. These data are collected in real time and stored in databases for detailed analysis, supporting in-depth evaluation of system behavior and identification of potential improvements.

From a hardware perspective, high-performance servers are deployed to host key AccesS components, such as the Data Base Layer (DBL) and the Digital Twin (DT). These servers are configured to handle multiple concurrent requests, providing robust data storage and processing capabilities essential for reliable system operation. A variety of test devices, including sensors, actuators, and smart home appliances, are used to emulate real-world conditions within the nZEB Smart House, further enhancing the realism and relevance of the testing environment.

Networking infrastructure is a critical element of the test setup. High-speed routers, switches, and related networking equipment ensure reliable and low-latency connectivity among all components of the AccesS Suite. This network configuration supports efficient data exchange and communication, enabling integrated components to operate cohesively and accurately reproduce real-world data flows. Such connectivity is essential for comprehensive integration and system testing, allowing validation of interoperability and performance under realistic conditions.

On the software side, the test infrastructure relies on stable and secure operating systems, with GNU/Linux (Ubuntu 22.04) deployed across servers and selected test devices to ensure consistency, reliability, and security. Databases such as MongoDB and PostgreSQL are used to manage and store data generated and consumed by AccesS components, supporting efficient data handling and retrieval. Application servers and runtime environments, including Node.js and related technologies, are deployed to support the execution of AccesS applications and testing activities. This comprehensive hardware and software setup ensures that the AccesS Suite is rigorously evaluated with respect to functionality, performance, and security, supporting its readiness for successful deployment.

2.3.2 Template

In order to ensure a structured and consistent approach to document the planned tests, a comprehensive template was prepared, which is presented in Table 1. This template supports the systematic documentation of the different testing activities carried out within T7.3, including unit, integration, security, and system tests, in line with the overall testing strategy. It contains fields for key information such as the test identifier, description, type of test, related AccesS Suite components, testers and responsible parties, test objective, system configuration, test steps, input data, and obtained results. The use of this template enables clear and comprehensive documentation of the testing procedures and facilitates effective communication of the test plans and outcomes.

Table 1. Test Description Template.

Test Title	*Title of the test to be performed*			
Identifier	*Unique identifier of the test*			
Test Type	*E.g.: Integration, Unit, Security, System*			
AccesS Suite components	*E.g.: Energy comfortness tool, VUMs*			
Tester(s)	*E.g.: CERTH*			
Responsible	*E.g.: FORTH*			
Test Purpose	*Purpose of the experiment and methodology to be followed*			
Configuration	*Configuration needed per component, concerning:* • *HW/SW and OS configuration* • *Network configuration; Etc.*			
Pre-test conditions	*The pre-conditions that need to be met by the different components, e.g., setting up IoT devices, MQTT brokers, etc*			
Test Sequence	**Step**	**Type**	**Description**	**Result**
	1	*Network*	*E.g., connecting IoT devices*	*Successful*
	2	*Functional*	*Setting up an MQTT broker…*	*In progress*
	…			
Input Data	*Data type and format*			
Results	*Screenshots*			

3 Test Execution, Results and Analysis

This section presents the execution and outcomes of the tests performed on the AccesS Suite, covering unit, integration, security, and system tests. Each subsection presents the main results obtained for each AccesS Suite component (including the AccesS Suite itself). Detailed test case documentation is provided in the Annex section for reference.

3.1 AccesS Suite

The AccesS Suite is the digital platform that combines all the developed tools and constitutes the main focal point of the project. At the time of writing this deliverable, all digital components have been integrated into the platform (shown in Figure 5), with the exception of Accessibility Assessment Tool (AAT), which has been tested only at a unit level.

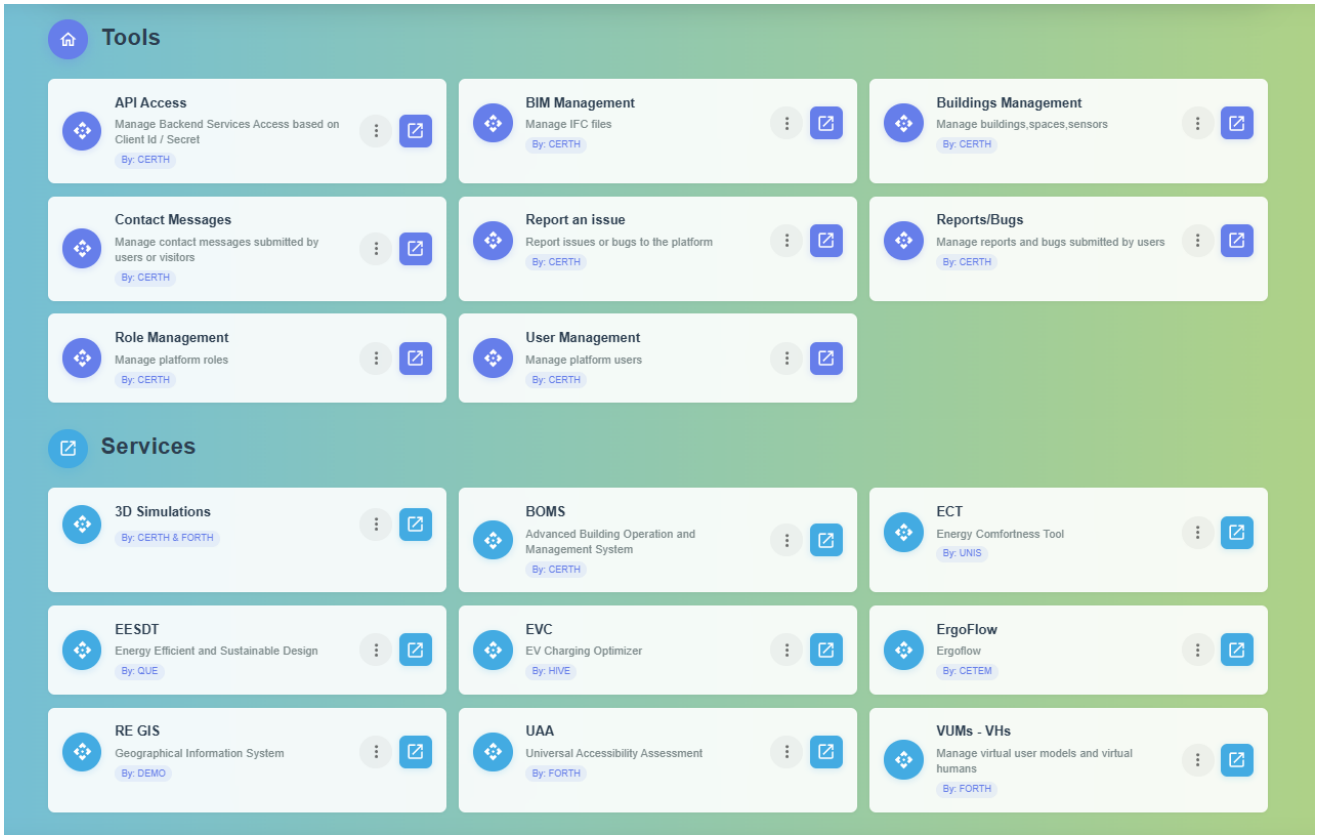


Figure 5. Screenshot from the AccesS Suite, showing the integrated tools and services.

3.1.1 Testing

For the testing purposes, the CERTH/ITI nZEB Smart House was added as an additional pilot through the Building Management System of the AccesS Suite, enabling the nZEB Smart House users to access all AccesS tools and services. All tests were conducted using the nZEB Smart House as the testbed. A test user was granted access to the nZEB Smart House building, and through Keycloak, roles and permissions were configured to provide access to all other AccesS tools and services.

3.1.1.1 Unit

The unit testing phase focused on verifying the correct operation of individual configuration elements and core functionalities of the AccesS Suite in isolation. The test assessed whether the essential configuration files, environment variables, and startup settings were correctly defined and loaded within the system environment. The testing was performed on a local development machine running Ubuntu 22.04.5, using Docker-based deployment with Keycloak for authentication, TimescaleDB as the database system, and a software stack based on Python, Node.js, and React.js. The objective was to ensure that the basic system components—including the Building Management module, user roles and permissions management, and user registration and invitation mechanisms—function correctly when executed independently of other integrated components.

The test execution followed a structured sequence of configuration and functional checks. First, the correctness and completeness of the .env configuration file were verified to ensure that all required Keycloak URIs and environment variables were properly defined. Subsequently, functional validation was performed on key AccesS Suite modules, including the Building Management system and the Roles and Permissions management functionality. Additional checks confirmed that user registration, login, and invitation workflows operated as expected. All test steps were completed successfully, indicating that the individual configuration files and internal system tools were properly configured and executed without errors under the defined test conditions.

The successful outcome of the unit tests demonstrates that the foundational configuration elements and core functionalities of the AccesS Suite operate reliably in isolation. This confirms that the internal modules are correctly initialized and that the authentication, role management, and user interaction mechanisms function as intended at the component level. Such validation is essential before proceeding to higher-level testing stages, as it ensures that individual system components meet their functional requirements and can serve as stable building blocks for integration and system testing.

3.1.1.2 Integration

The integration testing phase was conducted to verify the correct interaction between the AccesS Suite and its distributed services, ensuring that authenticated sessions, routing mechanisms, and data exchanges function reliably across independent server environments. The test focused on validating the orchestration between the AccesS Hub and external tools, as well as confirming secure and consistent communication between the suite applications and the central database. The testing environment was configured on Ubuntu 22.04.5 and included Keycloak for authentication, TimescaleDB for data storage, and application components developed using Python, Node.js, and React.js. Network communication between the AccesS Hub and the individual services was managed through Nginx proxy rules configured to route requests to the appropriate tool servers.

The test execution involved verifying both routing and data communication processes. First, routing functionality was evaluated by accessing an external tool (e.g., BOMS) through the AccesS Suite interface and confirming that the Nginx proxy configuration correctly redirected the request to the tool's dedicated server. The test confirmed that the routing mechanism functioned as expected and that authenticated sessions were preserved when accessing services hosted on different servers. Subsequently, communication between the AccesS applications and the TimescaleDB database was validated by examining typical user-driven workflows. These included the execution of queries, updates, and database transactions initiated through the suite's interface. The results confirmed that requests were correctly processed by the database and that the responses were accurately returned and reflected within the AccesS Suite applications.

The successful completion of the integration tests indicates that the system components can effectively interact within the distributed architecture of the AccesS Suite. The results demonstrate that the reverse proxy configuration, authentication persistence, and database communication

mechanisms are correctly implemented, enabling seamless interaction between the hub, external services, and the backend database. This level of interoperability is essential for ensuring reliable operation of the suite when multiple tools and services are deployed across separate servers.

3.1.1.3 Security

The security testing phase aimed to validate the authentication and authorization mechanisms of the AccesS Suite and to ensure that user access control is consistently enforced across the platform. In particular, the test focused on verifying the correct implementation of Single Sign-On (SSO) and Single Log-Out (SLO) mechanisms, as well as the effectiveness of Role-Based Access Control (RBAC) in restricting access to system functionalities and building-level data. The tests were conducted in an environment running Ubuntu 22.04.5, with Keycloak used as the authentication and identity management service and TimescaleDB serving as the database. The software stack included Python, Node.js, and React.js, while secure communication was ensured through SSL-enabled nodes.

The testing procedure involved a sequence of security checks across the distributed components of the AccesS Suite. Initially, the SSO functionality was validated by logging into the AccesS Hub and navigating to services hosted on different servers without requiring additional authentication. The system successfully maintained the authenticated session across these services, confirming proper token-based authentication using Keycloak. The logout process was subsequently tested through the execution of a Single Log-Out operation from the Hub, which correctly terminated active sessions across all connected services. Additional checks confirmed that user roles configured through the user interface were properly stored in the database and applied by the system. Finally, access control enforcement was tested by attempting to retrieve data associated with an unauthorized building entity. The system correctly blocked the request, confirming that RBAC policies were effectively applied and that users could only access resources for which they had explicit authorization.

The results indicate that the AccesS Suite successfully enforces centralized authentication and distributed authorization across its components. The correct implementation of SSO improves usability by allowing seamless navigation between services hosted on different servers, while the SLO functionality ensures consistent session management and enhanced security. Furthermore, the RBAC validation confirms that access to sensitive data is appropriately restricted based on user roles and building-level permissions, thereby supporting secure multi-user operation within the platform.

3.1.1.4 System

The system testing phase was conducted to evaluate the overall functionality and resilience of the AccesS Suite when operating as a fully integrated ecosystem. The objective of this test was to verify that users, specifically a Facility Manager, can seamlessly navigate and interact with the various services of the platform, even when these services are distributed across multiple servers. The testing environment was deployed on Ubuntu 22.04.5, using Keycloak for authentication and identity management, TimescaleDB for database operations, and application components developed with Python, Node.js, and React.js. The system architecture included multiple service nodes, each hosting different AccesS components, reflecting the distributed nature of the platform.

The execution of the system test included two main evaluation steps. First, an end-to-end workflow was performed to validate the overall user interaction with the platform. This workflow involved a complete sequence of actions, including user login through the AccesS Hub, navigation between different services and tools hosted on separate servers, and final logout from the platform. The results confirmed that the user could successfully access and navigate the ecosystem without interruptions, demonstrating the correct integration of authentication, service routing, and application functionality across the distributed architecture. The second step assessed system resilience by simulating a failure in one of the tool servers. The objective was to verify whether the remaining services of the AccesS Suite could continue operating without disruption. The results showed that the rest of the platform

remained accessible and operational, indicating that the system architecture is capable of isolating service-level failures without affecting the entire ecosystem.

3.1.2 Assessment

Based on the test results, the AccesS Suite demonstrates a stable and reliable configuration and deployment setup for the tested environment. The successful execution of security assessments confirms that the current authentication and authorization mechanisms are robust and effective. The successful results of the system testing demonstrate that the AccesS Suite can support full end-to-end user workflows within a distributed infrastructure. The platform allows users to interact with multiple services through a unified interface while maintaining stable operation even when individual components experience failures.

For the second and final phase of the inclusive toolbox testing, integration and system testing for the AccesS Suite could be expanded to include more demanding conditions, such as high user loads, concurrent requests, network latency, and extended operation periods, in order to better assess system behavior under realistic scenarios.

3.2 Energy Efficient & Sustainable Design Tool

The Energy Efficient and Sustainable Design Tool (EESDT) is a BIM-based digital solution that supports energy-efficient, smart, and sustainable building renovations. It enables users to evaluate and compare renovation and smartification scenarios by combining BIM data, energy performance simulations, and smart readiness assessments. The tool integrates key backend components, including the IFC Model Checker, BIM-to-BEP Simulator, BIM-to-SRI Calculator, and a Decision Support Engine, to validate models, simulate performance, assess smart readiness, and rank scenarios based on multiple criteria. Through a web-based interface, users can explore building models, configure scenarios, and analyze results via interactive dashboards.

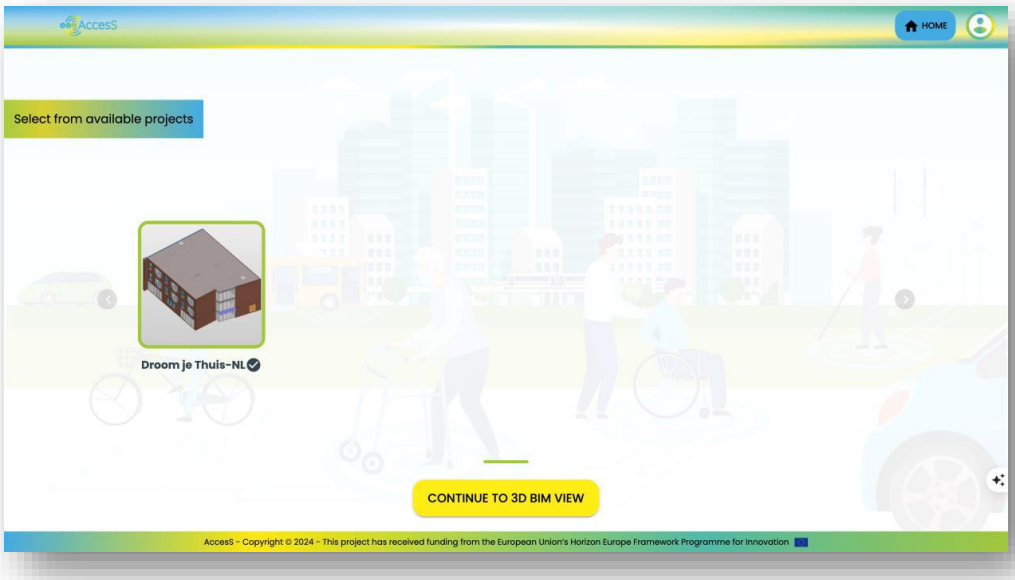


Figure 6. Screenshot from EESDT, showing the project selection interface.

3.2.1 Testing

3.2.1.1 Unit

The Energy Efficient and Sustainable Design Tool (EESDT) has been validated at a unit testing level, focusing on the correct operation of its core functional components and internal workflows. The objective of this testing phase was to ensure that the individual modules of the tool perform as expected in isolation, supporting the overall goal of enabling data-driven decision-making for building renovation strategies. The EESDT is a BIM-based decision-support environment that processes openBIM Industry Foundation Classes (IFC) models, integrates climate data and renovation measures, and performs Building Energy Performance (BEP) simulations and Smart Readiness Indicator (SRI) assessments to derive key performance indicators (KPIs).

The testing activities targeted the main functional components of the EESDT, including the IFC Model Checker, the Baseline Scenario Generator, the What-if Scenarios Generator, the BIM-to-BEP Simulator, the BIM-to-SRI Calculator, and the Decision Support Engine. The validation confirmed that IFC files can be successfully imported and processed, and that the IFC Model Checker correctly verifies model completeness and consistency against predefined data requirements using IDS-based validation rules. The Baseline Scenario Generator was tested to ensure that it correctly extracts building metadata, retrieves appropriate weather data, and prepares simulation-ready inputs. The BIM-to-BEP Simulator and BIM-to-SRI Calculator were also validated, confirming that energy performance simulations and smart readiness assessments are executed correctly and produce structured outputs in JSON format. Furthermore, the What-if Scenarios Generator was verified to correctly generate renovation scenarios based on selected measures from the ECM and SR catalogue, ensuring that alternative configurations are consistently created and processed through the same simulation pipeline as the baseline scenario.

In addition, the Decision Support Engine was tested to confirm that it correctly aggregates simulation outputs and applies multi-criteria evaluation methods to rank renovation scenarios. The internal data flow between components was validated, ensuring that outputs from one module are correctly used as inputs by subsequent modules, enabling a coherent workflow from model validation to decision support. Input data used during testing included IFC models exported from BIM tools, EnergyPlus weather files, and structured datasets describing renovation measures and smart technologies. The results demonstrated that all core components operate correctly at an individual level, producing consistent and valid outputs aligned with their intended functionality.

3.2.1.2 Integration & Security

Integration testing of the Energy Efficient & Sustainable Design Tool (EESDT) focuses on ensuring that the service correctly operates within a distributed system where authentication is centrally managed through Keycloak. In this architecture, users authenticate once via the Access Suite and receive an access token, which is then used to access the EESDT without additional login steps. Since authentication is handled externally, the responsibility of the EESDT is to properly validate and enforce the use of these tokens for all incoming requests.

A key aspect of the integration testing process is verifying that the EESDT correctly accepts and validates access tokens. This includes checking the token's integrity, signature, expiration time, and embedded claims such as user identity and roles. Only valid tokens should allow access to the system's functionality. Complementary to this, tests are performed to ensure that invalid scenarios are handled securely, including cases where tokens are missing, malformed, expired, or tampered with. In all such situations, the system must deny access and return appropriate responses, such as HTTP 401 Unauthorized, ensuring that no unauthorized interaction with the service is possible.

Another important focus is the handling of token expiration. Integration tests simulate scenarios where a token expires during usage, verifying that the EESDT immediately revokes access and consistently

enforces session validity. This guarantees that users cannot continue interacting with protected resources beyond the allowed authentication timeframe. Where applicable, tests also validate that the service correctly interprets role-related information contained within the token and enforces authorization rules accordingly.

In addition to authentication, integration testing also covers fine-grained authorization related to building-level access. Specifically, tests verify that users can only access data, simulations, and Building Information Management (BIM) files associated with the buildings for which they have been granted permissions. This involves validating that the EESDT correctly filters and restricts API responses based on the user's assigned building access rights, as defined either within the token claims or retrieved from the system. Scenarios include attempting to access authorized buildings (which should succeed) and unauthorized buildings (which must be rejected with appropriate responses, such as HTTP 403 Forbidden). These tests ensure strict data isolation between users and prevent unauthorized visibility of building-related information.

Furthermore, integration testing includes secure access to BIM files uploaded to the central hub and retrieved via its API. The tests ensure that BIM files can only be accessed when a valid access token is provided and when the user has explicit permission for the corresponding building. Requests made with expired, invalid, or insufficiently privileged tokens must be denied, preventing unauthorized access to sensitive building data.

Finally, integration tests validate the reliability of BIM file retrieval and processing. This includes verifying correct system behavior when files are unavailable, incorrectly referenced, or when the central hub returns errors. The EESDT must handle such cases gracefully, providing meaningful error responses without compromising system stability. Once BIM files are successfully retrieved, tests ensure that they are complete, uncorrupted, and compatible with the tool's processing mechanisms, allowing them to be used effectively for energy efficiency and sustainability analysis.

Overall, these integration tests ensure that the EESDT not only enforces authentication correctly but also applies strict, fine-grained authorization controls, guaranteeing that users can only access the data and building resources they are explicitly permitted to use, thereby ensuring both security and data integrity across the system.

3.2.2 Assessment

At the time of writing this deliverable, the EESDT has been tested only at a unit level and therefore, the current validation focuses exclusively on the standalone operation and internal consistency of the tool. Integration testing within the AccesS ecosystem, including interoperability with other components and user access through the platform, will be addressed in subsequent phases and documented in the next deliverable.

3.3 Energy Comfortness Tool

The Energy Comfortness Tool (ECT) is a software tool that provides a comprehensive analysis of human well-being and building efficiency by integrating Machine Learning and Building Information Modelling (BIM) with Building Energy Modelling (BEM).

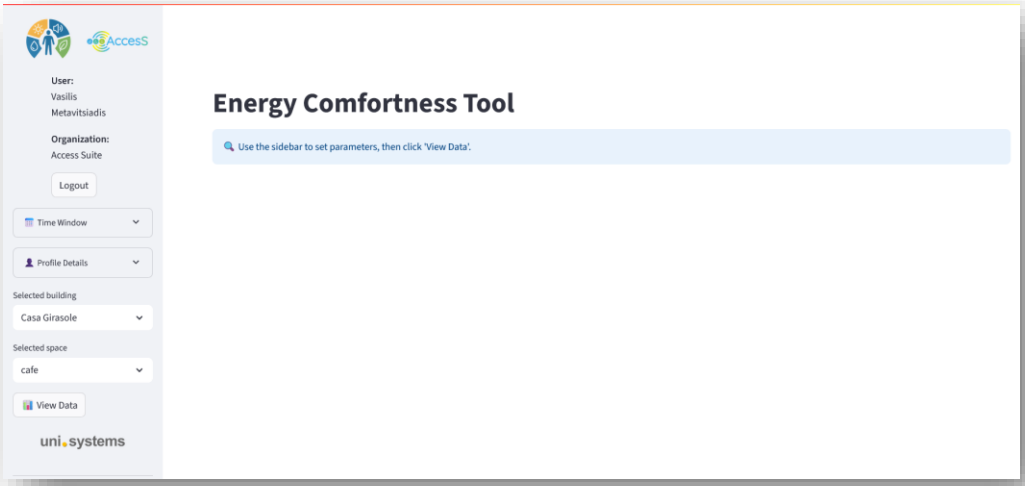


Figure 7. Screenshot from ECT simulation dashboard's starting screen.

3.3.1 Testing

3.3.1.1 Unit

This functional unit test aimed to verify that the Energy Comfortness Tool (ECT) correctly adapts comfort calculations according to different occupant profiles. The objective was to confirm that modifying user characteristics, such as age, gender, clothing insulation levels (CLO), metabolic rates (MET), and visual capability—appropriately influences the calculated thermal and visual comfort indicators and the resulting overall comfortness classification. The test was conducted using the ECT with a dataset containing thermal and visual environmental parameters, ensuring that the same environmental conditions were applied to multiple occupant profiles to isolate the effect of user-specific parameters on the results.

The testing procedure involved selecting two different user profiles within the system: a baseline profile (Adult-M) and an alternative profile representing an elderly female (Elderly-F) user with visual impairment. Initially, the Adult-M profile was selected and the thermal and visual comfort calculations were executed using the provided environmental dataset. This step generated baseline outputs, including PMV/PPD values for thermal comfort and a corresponding visual comfort score. Subsequently, the profile was changed to the Elderly-F profile, which incorporates modified parameters such as higher clothing insulation levels and visual impairment adjustments within the visual comfort model. The system successfully recalculated the comfort indicators using the same environmental data. The results showed higher predicted thermal dissatisfaction (higher PPD, indicating a colder perceived environment) and a reduced visual comfort score, reflecting the personalized characteristics of the second profile. As a result, the overall comfortness classification could shift to a lower comfort class (e.g., from B to C), demonstrating the impact of user-specific parameters on the evaluation.

The test results confirm that the Energy Comfortness Tool correctly incorporates occupant profile characteristics into the comfort evaluation process. By producing different comfort assessments under identical environmental conditions for different user profiles, the tool demonstrates its ability to support personalized comfort analysis. This functionality is particularly important for the AccesS framework, as it allows the system to account for diverse occupant needs—including those related to aging or accessibility considerations—when evaluating building comfort performance.

3.3.1.2 Integration & Security

The integration and security testing of the Energy Comfortness Tool (ECT) aimed to verify its correct operation when deployed as part of the integrated AccesS Suite environment. The primary objective was to validate the interaction between the ECT and the central AccesS infrastructure, including authentication via the Keycloak identity management system and automated data retrieval from the AccesS Data Hub. These tests evaluated both the secure access control mechanisms and the interoperability of the ECT with the suite's API services. The integrated version of the tool was deployed in a containerized environment (Docker) and connected to the AccesS authentication service and API gateway, enabling the testing of real user workflows under conditions representative of the operational platform.

The first test (ECT-UNIS-INT-001) focused on authentication and secure access management. When accessing the ECT landing page without prior authentication, the user was correctly redirected to the central AccesS Keycloak login interface. After entering valid credentials, the system successfully authenticated the user and redirected them back to the ECT dashboard. The dashboard correctly displayed the authenticated user identity and allowed navigation to tool functionalities, such as the Thermal Comfort tab. During this process, the system automatically loaded relevant building and space data from the AccesS infrastructure, confirming that authenticated sessions and permissions were properly managed. A negative scenario was also tested by logging out and attempting to access the ECT directly through its URL, which again triggered a redirection to the AccesS authentication service, demonstrating correct enforcement of authentication policies and preventing unauthorized access.

The second test (ECT-UNIS-INT-002) validated the automated ingestion of pilot data from the AccesS Data Hub through the platform's API services. After authentication, the tester accessed the Thermal Comfort tab and selected a specific building and space from the Droom je Thuis pilot case. The system successfully retrieved environmental sensor data (e.g., temperature, PM2.5, CO_2) and building metadata from the AccesS Data Hub for the selected period (July–September 2025). The retrieved data were automatically processed and visualized within the tool through time-series charts and comfort class distributions. The metadata displayed in the user interface, including building identifiers, space identifiers, and occupant profiles, were verified to match the records stored in the AccesS platform. A negative test case was also executed by selecting a non-existing space identifier, which correctly triggered an API response indicating that no data were available for the requested space. This behavior confirms that the tool handles invalid requests in a controlled and transparent manner.

3.3.1.3 System

This system level test aimed to verify the consistency of the Energy Comfortness Tool calculations between its standalone implementation (ECT v1) and the integrated version deployed within the AccesS Suite (ECT v2). The objective was to confirm that both versions rely on the same core calculation engine and therefore produce identical results when provided with the same input data and configuration parameters. This validation is important to ensure that the integration of the tool into the AccesS ecosystem does not alter the underlying algorithms or introduce deviations in the comfort evaluation process.

The testing procedure involved executing the same simulation workflow in both tool versions using an identical dataset. The dataset, consisting of time-series environmental data (e.g., temperature, relative humidity, CO_2 concentration) and the corresponding occupant profile information, was exported from the AccesS platform or obtained from a reference dataset such as the nZEB Smart House example. Initially, the dataset was imported into the standalone ECT v1 environment running locally in Docker, and the simulation was executed to calculate the thermal and visual comfort indicators and the overall comfortness classification. The resulting outputs, including comfort class distributions and numerical indices such as PMV and PPD, were recorded. Subsequently, the same dataset was processed in the

integrated ECT v2 version through the AccesS Suite interface, using the same occupant profile (e.g., Adult-M), time period, and weighting parameters for the overall comfortness calculation.

The comparison of results between the two environments confirmed that both versions produced identical outputs. The distribution of comfort classes, the overall comfortness score, and the calculated comfort indices showed zero deviation between the standalone and integrated implementations. These results demonstrate that the integration of the ECT within the AccesS Suite preserves the integrity of the underlying comfort calculation models and that the shared core computation module operates consistently across different deployment environments.

3.3.2 Assessment

The successful test outcomes demonstrate that the Energy Comfortness Tool (ECT) is fully operational, maintaining complete computational consistency and identical analytical results across both standalone and integrated versions. By successfully integrating with the AccesS Suite infrastructure, the tool supports centralized authentication, secure access, and automated data exchange via the AccesS Data Hub, effectively eliminating manual data uploads. Furthermore, the verified personalization functionality and seamless API-based ingestion ensure a scalable, transparent, and trustworthy user experience, confirming the ECT's readiness for deployment within the broader AccesS ecosystem.

3.4 ErgoFlow

ErgoFlow is comprised of the Evaluation Tool which is a web-based application that assesses the ergonomics and accessibility of seating and tables using a weighted scoring system based on international standards, and the ErgoFlow Library, which is a repository of pre-validated, 3D-ready furniture pieces, providing specific ergonomic scores, files, and reports for each item.

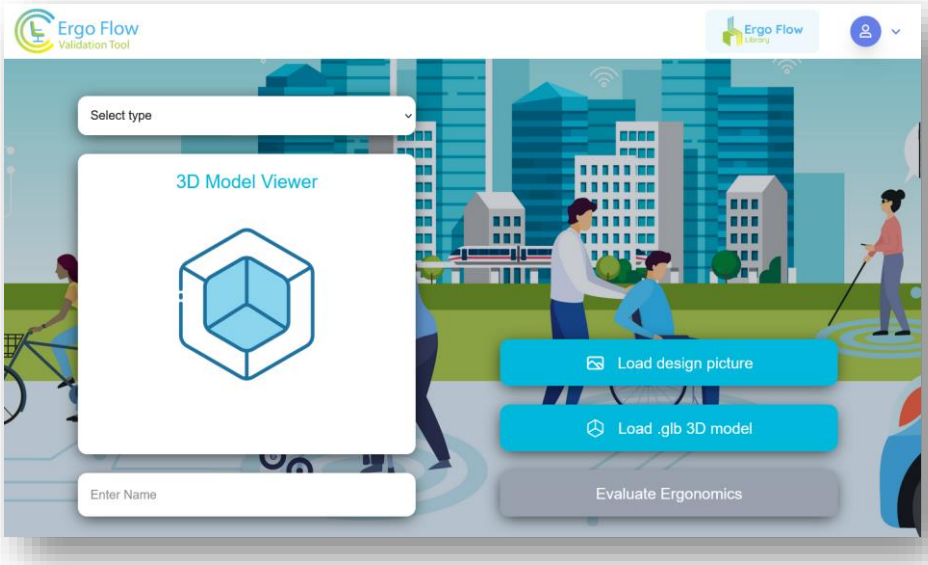


Figure 8. Screenshot of ErgoFlow's Evaluation Tool.

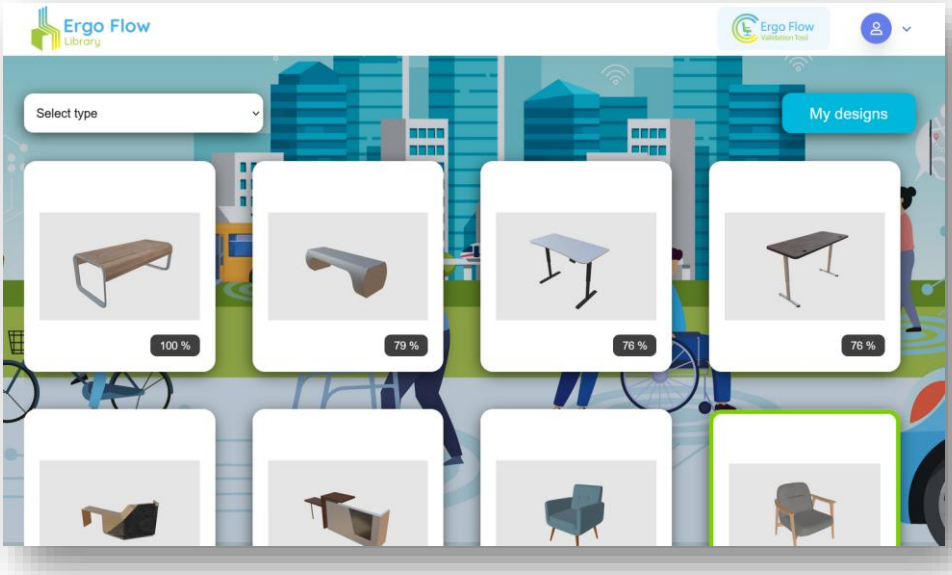


Figure 9. Screenshot of ErgoFlow Library.

3.4.1 Testing

Testing for the ErgoFlow system, included both the ErgoFlow Evaluation Tool and the ErgoFlow Library. The objective was to ensure that the platform can properly support the evaluation of furniture products in terms of ergonomics, comfort, and accessibility, while also verifying the correct storage and retrieval of evaluated models within the system library. The validation focused on confirming that the complete workflow, from model upload to evaluation and library integration, functions correctly within the AccesS Suite environment.

The system was tested through its web-based interface using a modern browser, with authentication performed through the AccesS Suite single sign-on mechanism. The ErgoFlow platform is implemented as a modular monolithic web application based on Next.js and React, operating on a Node.js runtime with MongoDB used for data persistence and Cloudinary for file storage. Secure HTTPS communication and REST API interactions between the frontend and backend components support the operation of the platform. Prior to the test, the system deployment, authentication services, database, and storage infrastructure were verified to be operational, while test user accounts and furniture models in the required formats were prepared.

After successfully accessing the platform through the browser and authenticating via the AccesS Suite, testers created a new evaluation project and selected a furniture type. Reference images and 3D furniture models in .glb format were uploaded to the platform and visualized within the integrated 3D viewer, allowing interaction with the model. The evaluators then manually introduced ergonomic and accessibility parameters according to the predefined criteria of the tool. Once the parameters were defined, the system executed the ergonomic evaluation process and produced a weighted score based on internationally recognized standards. The results were presented through parameter-level feedback and an overall ergonomic score. When the calculated score reached or exceeded the predefined threshold of 65%, the evaluated model was considered compliant and could be uploaded to the ErgoFlow Library. The final step of the test verified that compliant models and their associated evaluation data could be successfully retrieved from the library.

The validation used 3D models in .glb format and reference images in .jpg, .png, or .jpeg format, with a maximum file size of 10 MB. Additional inputs included manually defined parameters such as numeric values (e.g., seat height, depth, clearance dimensions), boolean indicators (e.g., adjustable height or non-slip surfaces), and categorical descriptors (e.g., furniture type or surface shape).

3.4.2 Assessment

All steps of the workflow were successfully completed, demonstrating that the system correctly supports model visualization, ergonomic evaluation, report generation, and library management.

3.5 Virtual Humans (VHs)

The Virtual Humans (VHs) component enables the creation of Virtual User Models (VUMs) and their embodiment as VHs for use in 3D simulation environments that allow for the evaluation of accessibility, usability, and safety in real-world environments, supporting in that way, the development of inclusive, adaptable, and equitable design solutions.

3.5.1 Testing

Testing for Virtual Humans (VHs) validated the overall functionality, integration, and security of the GUI editor used for the creation of Virtual User Models (VUMs) and the generation of corresponding VUM-based Virtual Humans (VHs). The objective was to ensure that the system supports the complete workflow for digital authoring of user models, generation of 3D virtual representations, and secure interaction with backend services within the AccesS Suite environment. The validation examined both the operational workflow of the editor and its resilience against unauthorized access and improper usage.

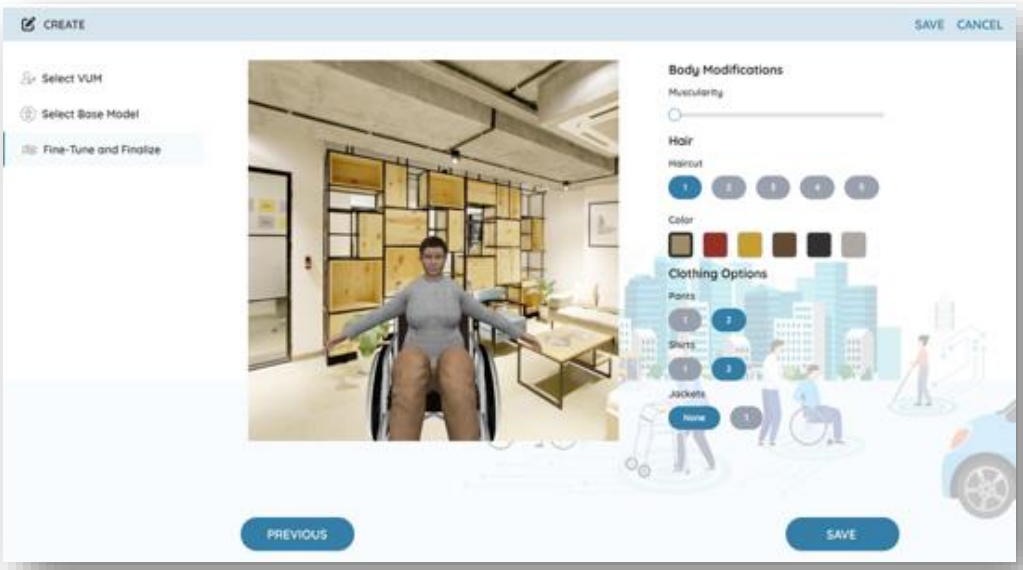


Figure 10. Screenshot from the VH creation within the Unit testing framework.

The testing environment consisted of a web-based frontend deployed on an externally accessible web server and a REST API deployed as a Docker container exposed through controlled endpoints. Supporting services included MongoDB for storing configuration data and metadata, Neo4j for managing the ontology representing aging individuals and people with disabilities, and MinIO object

storage for generated VH assets such as 3D files. Authentication and authorization were handled through Keycloak, enabling single sign-on across the AccesS Suite. The system was deployed using Docker containers connected through a private internal network, with only the frontend and REST API accessible externally, while all other services remained isolated. Secure communication between services was enforced and HTTPS was enabled for external access.

The system and integration testing evaluated the full operational workflow of the GUI editor. After authenticating through the Keycloak single sign-on mechanism, users accessed the editor and created a new VUM by entering demographic information, selecting disability attributes, and fine-tuning the relevant metrics. The system correctly applied predefined default values and disability effects to the model parameters. A Virtual Human was then generated from the selected VUM, allowing the user to choose a base model and verify parameter-driven changes such as age, height, weight, mobility aids, and posture. Additional visual characteristics, including clothing, hair, and accessories, were configured before finalizing the model. The generated Virtual Human was successfully exported as a .glb file suitable for use in the AccesS Suite 3D Scenarios Simulator. The integration between the graphical interface and the backend REST services functioned correctly throughout the process, while modifications to an existing VUM were correctly propagated to the generated VH after regeneration. The test also confirmed that VUM and VH entities could be edited and deleted through the interface, with the corresponding updates correctly reflected in the backend systems.

Security validation was conducted in parallel to assess the robustness of the system against unauthorized access and malicious interactions. Multiple scenarios were tested, including invalid login attempts, direct access to the GUI without authentication, attempts to access VUMs belonging to other users, and unauthorized API requests. In all cases, access was correctly denied by the authentication and authorization mechanisms. Additional tests included the submission of malformed or unexpected input payloads to the API, which were safely rejected without affecting system stability. These checks confirmed that the Keycloak-based authentication, secured API endpoints, and internal service isolation effectively protect the system against common security threats.

3.5.2 Assessment

The results demonstrated that the GUI editor reliably supports the creation, modification, and deletion of VUMs, correctly applies disability-related parameter effects, and generates consistent Virtual Humans that accurately reflect the selected user model parameters. The exported 3D models were generated successfully and were compatible with the AccesS Suite 3D simulation environment. Security checks confirmed that unauthorized access attempts and malformed inputs are properly handled, ensuring that the system maintains data integrity and secure operation within the AccesS platform.

3.6 3D Simulation Environments

The 3-D Simulation Environment component, consists of the 3D Simulator and the Semantic Annotation Tool for 3D Environments (SANTO), which transforms raw architectural data into interactive environments by assigning detailed semantic, kinematic, and structural information. This enhanced workflow features improves IFC-derived data support and granular control over mechanical parameters.

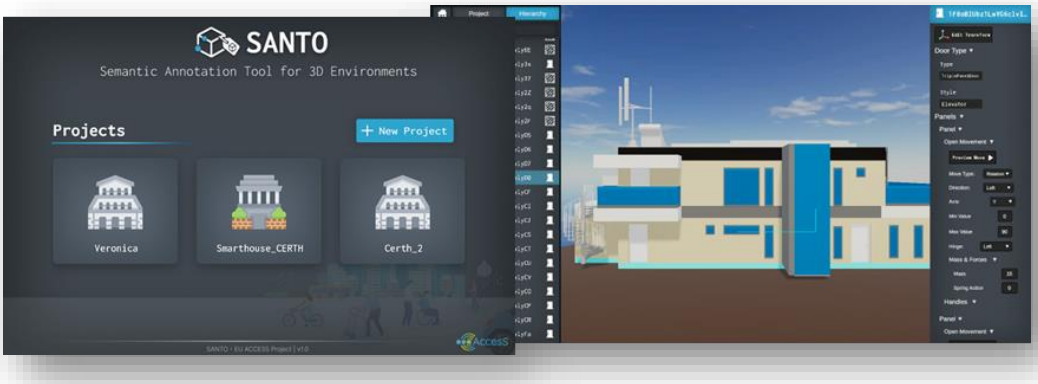


Figure 11. Screenshots from SANTO Tool.

3.6.1 Testing

Testing for 3-D Simulation Environments validated its correct operation and interoperability within the AccesS Suite, focusing on both the internal dynamic simulation behavior and the integration of the simulator with the Virtual User Models (VUMs)/Virtual Humans (VHs) toolchain and the SANTO application. The objective was to ensure that the simulation environment correctly interprets user models and environmental data, performs reliable biomechanical calculations, and communicates effectively with the external services providing the required input data.

The unit-level validation focused on the correctness of the dynamic simulation engine implemented in the 3D simulator. The test evaluated whether the biomechanical calculations produced by the simulation correspond to values reported in the scientific literature for human movement. The simulator was executed using input data provided in JSON format from both the VUMs module and the SANTO application, representing user characteristics and environmental configuration respectively. During the test, the simulation engine calculated the forces applied on human joints while performing typical activities such as normal walking and ascending or descending stairs. The resulting outputs were compared with reference values from existing literature on human biomechanics. The comparison confirmed that the calculated joint forces, particularly those affecting the knee joints, were consistent with the expected ranges reported in published studies, demonstrating that the dynamic simulation algorithms produce valid results.

The integration test examined the interoperability between the 3D simulation application, the VUM editor, and the SANTO system. Prior to the test, an IFC model was edited and prepared within the SANTO application, while corresponding VUM data describing user characteristics was generated through the VUM editor. The simulation environment then accessed these datasets through the corresponding backend services. The test verified successful connectivity with the backend services developed by FORTH and confirmed that the simulator correctly received the required JSON-formatted data from both systems. In addition to verifying connectivity, the test ensured that the structure and format of the exchanged data matched the expected API specifications, allowing the simulator to correctly interpret the environmental and user parameters necessary for running the simulation.

3.6.2 Assessment

The results of these tests confirmed that the 3D Simulation Environment operates correctly both at the computational level and at the system integration level. The dynamic analysis module generates biomechanical outputs consistent with established literature, confirming the validity of the simulation calculations. At the same time, the integration tests demonstrated reliable communication between

the simulator and the external modules providing environmental and user data, ensuring that the simulator can operate as part of the wider AccesS Suite ecosystem.

3.7 Adaptive Building Operation & Management Systems

BOMS is an intelligent decision support tool for real-time monitoring and adaptive optimization of building operations. The tool addresses increasing demands for energy efficiency, occupant comfort, and operational flexibility by leveraging advanced technologies such as smart sensors, IoT devices, and deep learning models. These capabilities enable buildings to respond dynamically to changing conditions and user needs.

For the testing, the nZEB Smart House was added as an additional pilot through the Building Management System of the Access Suite, enabling nZEB Smart House users to access all AccesS tools and services. All tests were conducted using the nZEB Smart House as the testbed. The data collected from the nZEB Smart House served three purposes: to test the connection with the database, ensuring proper data retrieval and storage; to train the forecasting engine and the decision support tool using load and PV measurements, ensuring accurate forecasts and recommendations; and to populate the BOMS dashboard, allowing verification of the user interface and monitoring functions. Additionally, Droom je Thuis pilot site also served as a testing case.

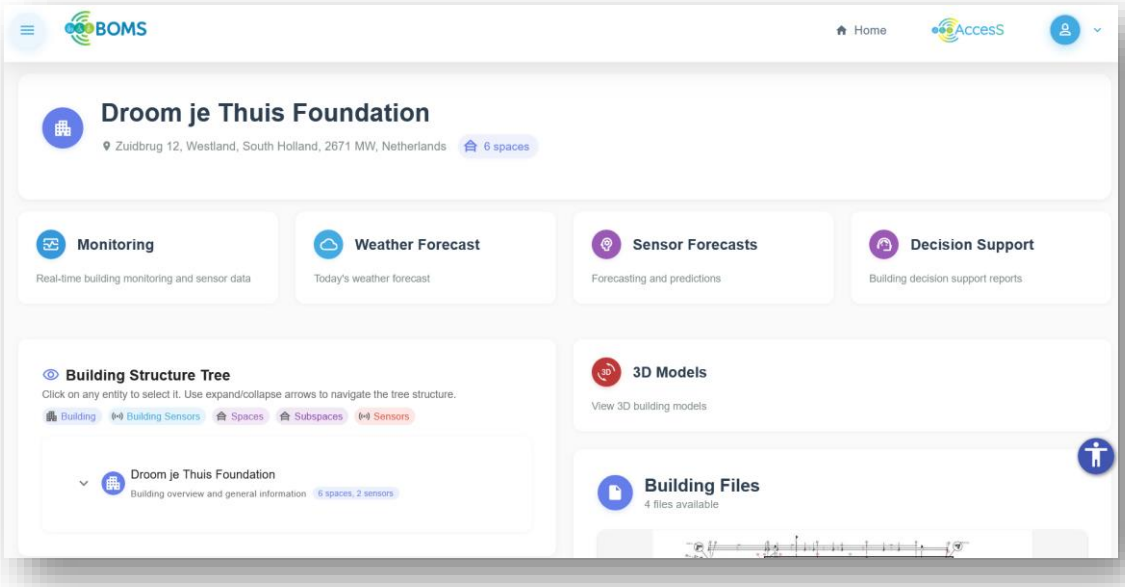


Figure 12. Screenshot from BOMS.

3.7.1 Testing

A comprehensive testing procedure was conducted to validate the functionality, integration, security, and overall operational workflow of the Building Operation Management System (BOMS) within the AccesS Suite. The objective was to ensure that the internal components of BOMS, including the RE-GIS, operate correctly in isolation, interact reliably with each other and the AccesS infrastructure, enforce secure access control, and support the complete end-to-end workflow expected by users.

3.7.1.1 Unit

The unit testing phase focused on verifying the correct operation of the individual BOMS components in isolation. The main modules evaluated included the forecasting engine, the decision support tool,

and the BOMS dashboard interface. Each module was executed using valid input parameters provided in structured formats such as JSON and numeric values. The tests confirmed that the internal functions of each module produced the expected outputs and behaved consistently under normal conditions. These tests were conducted in an environment running Ubuntu 22.04.5 with the BOMS software stack implemented in Python, Node.js, and React.js, ensuring that all required libraries and testing frameworks were properly configured. The results demonstrated that the individual functions of the forecasting engine, decision support tool, and dashboard components operated correctly and produced valid outputs when executed independently.

3.7.1.2 Integration

Integration testing then evaluated the interoperability of the BOMS components, including the RE-GIS, within the AccesS Suite ecosystem. The system was deployed in an environment where Keycloak provided authentication services and TimescaleDB served as the primary database for storing pilot data, forecasts, and recommendation outputs. During the test, users accessed BOMS through the AccesS Suite interface and authenticated via the Keycloak single sign-on mechanism. The BOMS dashboard successfully connected to the database and retrieved pilot data for monitoring purposes. Subsequently, the forecasting engine accessed the stored data, generated forecasts, and wrote the results back to the database. The decision support module then processed these forecasts and produced operational recommendations, which were also stored in the database. All components communicated successfully, confirming that the BOMS dashboard, forecasting engine, and decision support modules operate correctly as an integrated ecosystem.

3.7.1.3 Security

Security testing was performed to verify that BOMS enforces proper authentication, authorization, and session management through the Keycloak identity management service. The tests evaluated multiple security scenarios, including login attempts with valid and invalid credentials, attempts to access the system without authentication, verification of role-based access control policies, and validation of secure session handling mechanisms such as logout and session expiration. The system successfully denied unauthorized access attempts, enforced role restrictions, and maintained secure user sessions. These results confirmed that the authentication and access control mechanisms protect the system against unauthorized usage and ensure secure interaction with the platform.

3.7.1.4 System

Finally, system workflow testing validated the complete end-to-end operation of BOMS from the perspective of a typical user interacting with the AccesS Suite. The test simulated a normal operational scenario in which a user accessed the AccesS Suite platform, authenticated through Keycloak, and launched the BOMS application. Once inside the system, the user was able to retrieve and visualize pilot data from the database through the dashboard interface. The user then executed forecasting operations using the forecasting engine and obtained operational recommendations generated by the decision support module. The workflow concluded with the user logging out of the system. Each step of the workflow was completed successfully, demonstrating that BOMS supports the full operational cycle required for monitoring building data, generating forecasts, and producing actionable recommendations.

3.7.2 Assessment

The testing results confirm that the BOMS platform operates reliably at multiple levels: individual component functionality, integration within the AccesS Suite infrastructure, secure authentication and access control, and complete system workflow execution. This validation demonstrates that the BOMS

component is stable, secure, and capable of supporting building operation analysis and decision-making processes within the broader AccesS ecosystem.

3.8 EV Charging Optimizer

The EV Charging Optimizer is a modular tool designed to streamline scheduling and accessibility. It pulls various data streams together, using an OCPP connection to link directly with the operator's management system for real-time station metadata. The EV Charging Optimizer utilizes historical usage and grid metering data to predict demand and manage peak loads, ensuring charging schedules align with local grid constraints. The system also evaluates long-term scalability through growth indicators and prioritizes accessibility by ensuring that designated charging stations for disabled or elderly users are exempt from load-shifting measures.

3.8.1 Testing

This testing for the EV Charging Optimizer validated its correct operation, integration, and accessibility within the AccesS Suite environment. The objective was to ensure that the tool is properly integrated into the platform, supports secure user access through role-based authentication, processes real-world charging data, and provides accurate and up-to-date outputs through its dashboards. The validation also examined whether the optimizer executes its core functionality reliably on a recurring basis and correctly utilizes input data from field integrations.

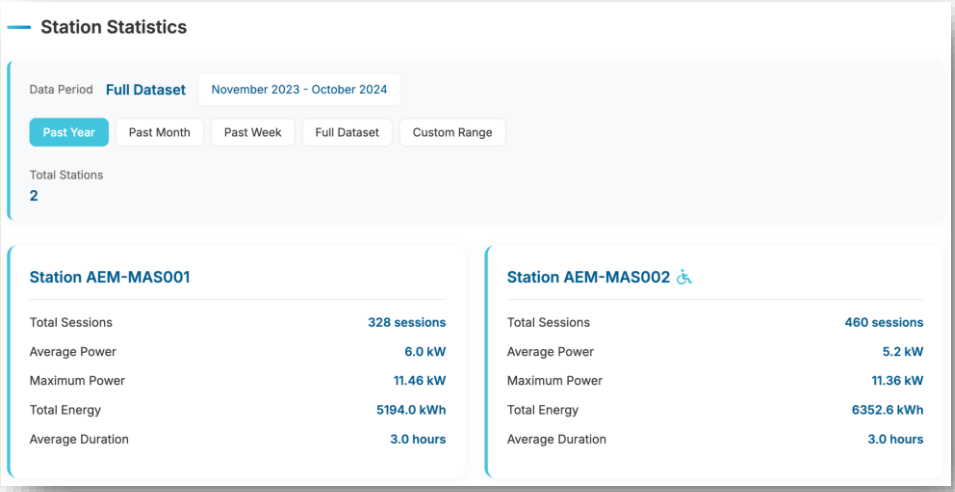


Figure 13. Screenshot from the EV Charging Optimizer's user interface, depicting characteristics of different charging stations.

The testing environment included a backend implemented in Python and a frontend based on HTML and JavaScript, with Plotly.js used for interactive data visualizations. Authentication and access control were handled through the Keycloak service, while secure HTTPS communication and API connectivity enabled data exchange between the optimizer and the AccesS Suite. Prior to testing, charging station data was made available through APIs, accessibility criteria were defined as input parameters, and Keycloak roles and authentication services were configured and operational.

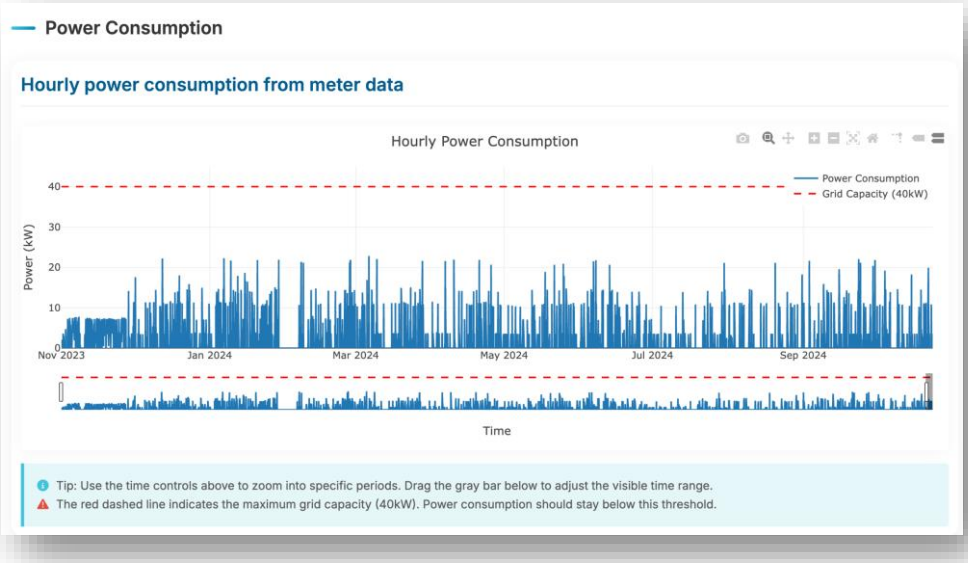


Figure 14. Screenshot from the EV Charging Optimizer's user interface, depicting hourly consumption.

The test procedure evaluated both system integration and functional behavior. Initially, the integration between the EV Charging Optimizer and the AccesS Suite was verified, confirming that the tool is accessible through the platform. User authentication via Keycloak was successfully tested, ensuring that access is granted based on predefined roles. The system then validated the availability and correct ingestion of charging station data received through external APIs. The core processing logic of the optimizer, implemented in Python, was executed and confirmed to run correctly, including its scheduled daily operation and data updates.

From a user interface perspective, the test verified that the generated results are correctly visualized through the available dashboards. Both the EV driver dashboard and the Charge Point Operator (CPO) dashboard displayed the expected data, confirming that processed information is accurately presented to different user roles. The input data used during testing included charging session datasets in JSON and CSV formats, containing fields such as timestamp, charging power, session duration, and station identifiers.

3.8.2 Assessment & Recommendations

The results demonstrated that the EV Charging Optimizer is fully integrated within the AccesS Suite, operates reliably with real-world data inputs, and provides accurate and accessible visual outputs. Authentication and role-based access control function correctly, data ingestion and processing are performed as expected, and the dashboards present the results effectively. This confirms that the tool is ready to support optimized EV charging management within the AccesS platform.

3.9 Accessibility Assessment tool

The large-scale Accessibility Assessment Tool is a digital platform that serves as a systematic tool for evaluating accessibility in physical structures and built environments. It enables assessments that follow standard guidelines and facilitates the implementation of the AccesS accessibility assessment framework. By utilizing both simulation-based tests and structured evaluation methods, the platform helps stakeholders analyze design options and pinpoint potential barriers during the planning and

review phases.

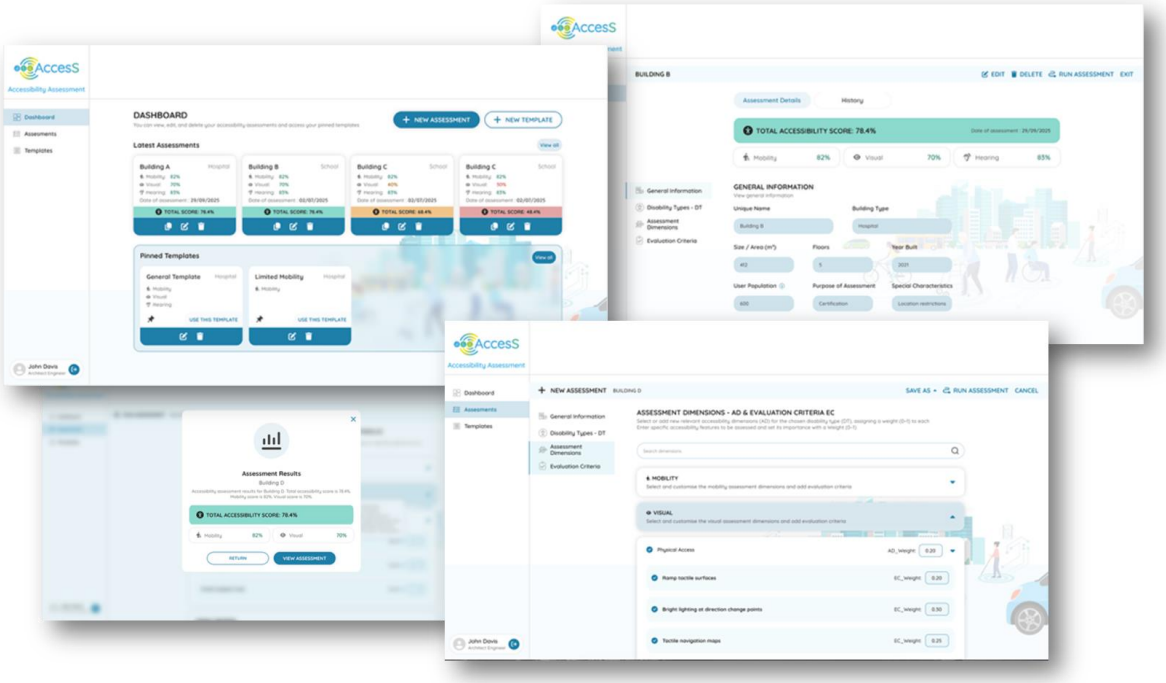


Figure 15. Screenshots from the Accessibility Assessment tool's user interface.

3.9.1 Testing

Since the Large-Scale Accessibility Assessment Tool has been fully integrated into the AccesS Suite in M22, the tool's testing focused on validating both the correctness of its core assessment workflow and the robustness of its security. The objective was to ensure that the tool reliably performs accessibility evaluations based on the defined assessment scheme, while also protecting against unauthorized access and improper use.

3.9.1.1 Unit

The unit testing phase examined the end-to-end assessment workflow at the application level, ensuring that all internal components function cohesively. The Accessibility Assessment Tool is implemented as a web-based system with a frontend accessible via a browser and a backend REST API deployed in a Docker environment. MongoDB is used to store the Accessibility Assessment Scheme and related configuration data, while MinIO handles storage of generated assessment reports. During testing, users accessed the platform and authenticated successfully, after which the frontend retrieved the assessment scheme data from the backend. Users were then able to define weights corresponding to disability types, assessment dimensions, and evaluation criteria. The system executed the scoring process by applying the underlying mathematical model and produced key outputs such as the total impact score and overall building accessibility score. The generated assessment reports were stored in the object storage service and retrieved through the frontend interface. Repeated execution of the same assessment with identical inputs confirmed the consistency and traceability of results, demonstrating that the tool produces stable and repeatable outputs. Overall, the system components, including frontend, API, database, and storage, operated cohesively, and the scoring calculations were performed correctly.

3.9.1.2 Security

The security testing phase evaluated the system's ability to prevent unauthorized access and ensure secure handling of user data and system resources. The tests included scenarios such as login attempts with invalid credentials, attempts to access the application interface without authentication, unauthorized access to resources belonging to other users, and direct unauthorized API requests. In all cases, the system correctly denied access through the Keycloak-based authentication and authorization mechanisms. Additional testing involved submitting malformed or unexpected input data to the API, which was safely rejected without affecting system stability. These checks confirmed that the AAT enforces proper access control, protects against data leakage, and maintains resilience against malformed or malicious inputs.

3.9.2 Assessment

The results of these tests demonstrate that the Accessibility Assessment Tool reliably supports the accessibility assessment workflow, correctly computes and stores evaluation results, and maintains consistency across repeated executions. At the same time, the system enforces secure authentication and access control.

The unit and security tests described above refer to the standalone operation of the Accessibility Assessment Tool component rather than its interaction within the full platform ecosystem. Integration testing with the AccesS Suite is planned for the next phase, and the corresponding validation results will be reported in the subsequent deliverable, where the interoperability and end-to-end operation of the Accessibility Assessment Tool within the platform will be assessed.

4 Conclusions

This deliverable presented the first version of the acceptance testing activities for the AccesS Suite, carried out for Task 7.3 “Inclusive Toolbox -Testing”. The testing process covered unit, integration, security, and system-level validation of the AccesS Suite and its components and services, with the objective of ensuring correct functionality, interoperability, and readiness prior to real-world deployment.

The results demonstrate that the AccesS Suite operates reliably across all tested layers. Unit testing confirmed that the components perform their intended functions correctly in isolation. Integration testing validated the seamless communication between components, including authentication via Keycloak, data exchange through APIs, and interaction with shared data infrastructures. Security testing confirmed that access control, authentication mechanisms, and protection against unauthorized actions are effectively implemented across the platform. System-level testing further demonstrated that complete end-to-end workflows can be executed successfully, ensuring that users can navigate and operate the AccesS ecosystem as a unified platform.

All components have reached a high level of integration within the AccesS Suite and demonstrated correct operation under realistic conditions and interoperability with related components while for the case of the large-scale Accessibility Assessment Tool, the tool has been delivered in M22 and therefore, currently it has been validated at unit level, while any additional testing will take place in the next phase.

Overall, the results of the AccesS Acceptance Tests confirm that the AccesS Suite has achieved a solid level of technical maturity, with stable functionality, effective integration mechanisms, and secure operation. The outcomes provide confidence in the platform’s readiness for further refinement, large-scale validation, and deployment, while the examination of these outcomes has identified areas for further improvement. D7.6 “AccesS Acceptance Tests v2”, the final version of this deliverable, will build upon these results by extending testing activities, completing the integration and system testing of all components, and track the changes that will be made towards the further optimization of the AccesS Suite.

ANNEX A: Tests

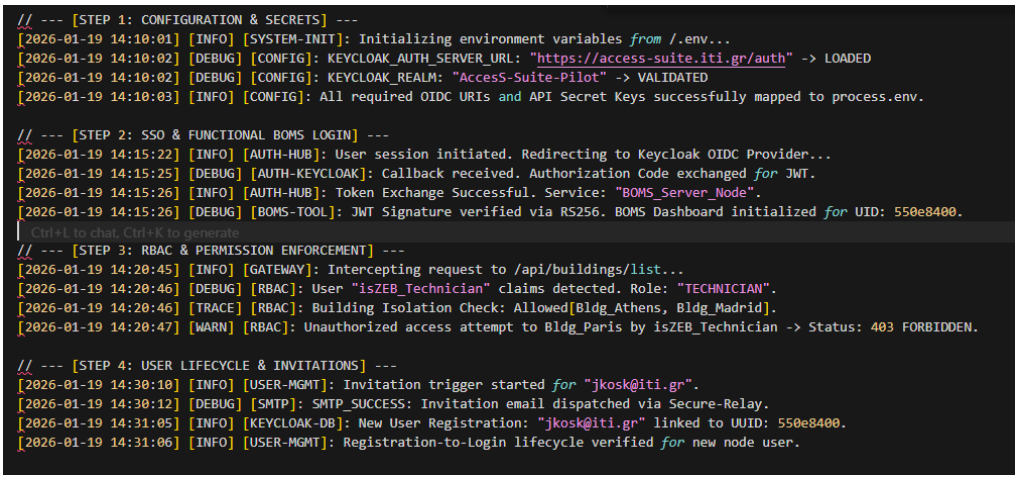
AccesS Suite

Table 2. AccesS Suite Unit Testing

Test Title	*AccesS Suite Unit Testing*
Identifier	*AccesS Suite -UT--1*
Test Type	*Unit Test*
AccesS Suite components	*AccesS Suite Core, Docker Configs, Environment Variables, Roles, Registration*
Tester(s)	*UNIS, CERTH, IsZEB*
Responsible	*UNIS, CERTH, IsZEB*
Test Purpose	*Verify that individual configuration files and startup scripts for the AccesS suite environment load correctly in isolation. Proper functionality of the internal AccesS suite tools (Building Management, Roles and Permissions management, User Invite).*
Configuration	*HW/SW and OS configuration* • *OS: UBUNTU 22.04.5* • *Authentication Service: Keycloak* • *Database: TimescaleDB* • *Programming Language: Python, Node.js, React.js* • *GPU: RTX 4080* *Local development machine Docker Engine installed; Config files (.env, YAML).*
Pre-test conditions	*Repository cloned; Ports 80/443 available; Environment variables defined.*

Test Sequence

Step	***Type***	***Description***	***Result***
1	*Config*	*Validate that the .env file contains all required Keycloak URIs*	*Successful*
2	*Functional*	*Validate the proper functionality of the Building Management system*	*Successful*
3	*Functional*	*Validate the proper configuration of User's Roles and Permissions*	*Successful*

	4	*Functional*	*Validate the proper registration, login and Invitation functionality.*	*Successful*
Input Data	*. env configuration files; YAML deployment manifests; user Roles*			
Results				

```
// --- [STEP 1: CONFIGURATION & SECRETS] ---
[2026-01-19 14:10:01] [INFO] [SYSTEM-INIT]: Initializing environment variables from /.env...
[2026-01-19 14:10:02] [DEBUG] [CONFIG]: KEYCLOAK_AUTH_SERVER_URL: "https://access-suite.iti.gr/auth" -> LOADED
[2026-01-19 14:10:02] [DEBUG] [CONFIG]: KEYCLOAK_REALM: "AccesS-Suite-Pilot" -> VALIDATED
[2026-01-19 14:10:03] [INFO] [CONFIG]: All required OIDC URIs and API Secret Keys successfully mapped to process.env.

// --- [STEP 2: SSO & FUNCTIONAL BOMS LOGIN] ---
[2026-01-19 14:15:22] [INFO] [AUTH-HUB]: User session initiated. Redirecting to Keycloak OIDC Provider...
[2026-01-19 14:15:25] [DEBUG] [AUTH-KEYCLOAK]: Callback received. Authorization Code exchanged for JWT.
[2026-01-19 14:15:26] [INFO] [AUTH-HUB]: Token Exchange Successful. Service: "BOMS_Server_Node".
[2026-01-19 14:15:26] [DEBUG] [BOMS-TOOL]: JWT Signature verified via RS256. BOMS Dashboard initialized for UID: 550e8400.
Ctrl+L to chat, Ctrl+K to generate
// --- [STEP 3: RBAC & PERMISSION ENFORCEMENT] ---
[2026-01-19 14:20:45] [INFO] [GATEWAY]: Intercepting request to /api/buildings/list...
[2026-01-19 14:20:46] [DEBUG] [RBAC]: User "isZEB_Technician" claims detected. Role: "TECHNICIAN".
[2026-01-19 14:20:46] [TRACE] [RBAC]: Building Isolation Check: Allowed[Bldg_Athens, Bldg_Madrid].
[2026-01-19 14:20:47] [WARN] [RBAC]: Unauthorized access attempt to Bldg_Paris by isZEB_Technician -> Status: 403 FORBIDDEN.

// --- [STEP 4: USER LIFECYCLE & INVITATIONS] ---
[2026-01-19 14:30:10] [INFO] [USER-MGMT]: Invitation trigger started for "jkosk@iti.gr".
[2026-01-19 14:30:12] [DEBUG] [SMTP]: SMTP_SUCCESS: Invitation email dispatched via Secure-Relay.
[2026-01-19 14:31:05] [INFO] [KEYCLOAK-DB]: New User Registration: "jkosk@iti.gr" linked to UUID: 550e8400.
[2026-01-19 14:31:06] [INFO] [USER-MGMT]: Registration-to-Login lifecycle verified for new node user.
```

Table 3. AccesS Suite Security Testing

Test Title	*AccesS Suite Security Testing*
Identifier	*AS-SEC--V1*
Test Type	*Security Testing*
AccesS Suite components	*All Components*
Tester(s)	*UNIS, CERTH, IsZEB*
Responsible	*CERTH, UNIS, IsZEB*
Test Purpose	*Validate that a single login on the Hub grants access to tools on different servers without re-authentication. Validate Role-Based Access Control (RBAC) across the Suite to ensure that user permissions strictly align with defined feature-sets and building-level data.*
Configuration	*HW/SW and OS configuration* • *OS: UBUNTU 22.04.5* • *Authentication Service: Keycloak* • *Database: TimescaleDB* • *Programming Language: Python, Node.js, React.js* • *GPU: RTX 4080* *Keycloak Realm with Clients for each tool; SSL enabled on all nodes.*

Pre-test conditions	*All Tool Servers registered as valid Redirect URIs in Keycloak.*			
Test Sequence	***Step***	***Type***	***Description***	***Result***
	1	*SSO*	*Login to AccesS Suite url and navigate on a AccesS-Suite Service that runs on different Server*	*Successful*
	2	*Logout*	*Perform Single Log-Out (SLO) on the Hub; verify session ends on the other server*	*Successful*
	3	*Settings*	*User Roles are configured and saved to Database through the UI*	*Successful*
	4	*Permissions*	*Attempt to fetch data from "Building B" using a user account only authorized for "Building A".*	*Successful*
Input Data	*User credentials; JWT Public Key (JWKS) URL.*			
Results				

```
2026-01-19 09:15:02 | INFO  | Test Started: AccesS Suite Communication & Access Control Validation
2026-01-19 09:15:05 | INFO  | Test Environment: Internal
2026-01-19 09:15:05 | INFO  | Test Type: Functional / Integration / Security

2026-01-19 09:15:10 | STEP 1 | SSO - Logging into AccesS Suite and navigating to a service on a different server
2026-01-19 09:15:12 | RESULT | Authentication completed successfully
2026-01-19 09:15:12 | RESULT | User session recognized across servers
2026-01-19 09:15:12 | RESULT | Target service loaded without errors
2026-01-19 09:15:12 | STATUS | PASS

2026-01-19 09:20:05 | STEP 2 | SLO - Performing Single Log-Out from the hub
2026-01-19 09:20:06 | RESULT | User session terminated on the hub
2026-01-19 09:20:06 | RESULT | Active session on secondary server invalidated
2026-01-19 09:20:06 | RESULT | Further access requires re-authentication
2026-01-19 09:20:06 | STATUS | PASS

2026-01-19 09:25:10 | STEP 3 | Settings - Configuring and saving user roles
2026-01-19 09:25:12 | RESULT | Role changes saved successfully
2026-01-19 09:25:12 | RESULT | Updates persisted in database
2026-01-19 09:25:12 | RESULT | Changes reflected immediately in system behavior
2026-01-19 09:25:12 | STATUS | PASS

2026-01-19 09:30:15 | STEP 4 | Permissions - Attempting to access unauthorized resource
2026-01-19 09:30:16 | RESULT | Access denied as expected
2026-01-19 09:30:16 | RESULT | No data leakage observed
2026-01-19 09:30:16 | RESULT | Proper error message displayed in UI
2026-01-19 09:30:16 | STATUS | PASS

2026-01-19 09:35:00 | INFO  | Test Completed: All steps executed successfully
2026-01-19 09:35:00 | STATUS | OVERALL PASS
```

Table 4. AccesS Suite Integration Testing

Test Title	*AccesS Suite Integration Testing*
Identifier	*AS-INT--V1*
Test Type	*Integration Test*

AccesS Suite components	*All Components*
Tester(s)	*UNIS, CERTH, IsZEB*
Responsible	*UNIS, CERTH, IsZEB*
Test Purpose	*Verify the efficient orchestration between the AccesS Suite and the external services, ensuring that authenticated sessions and data flows are maintained across independent server environments. Validate the proper and secure communication between the AccesS Suite and the Database.*
Configuration	*HW/SW and OS configuration* • *OS: UBUNTU 22.04.5* • *Authentication Service: Keycloak* • *Database: TimescaleDB* • *Programming Language: Python, Node.js, React.js* • *GPU: RTX 4080* *Network: Nginx proxy_pass rules configured for all Tool IPs.*
Pre-test conditions	*1. All external services (BOMS, etc.) are online and reachable via internal/external IPs.* *2. Keycloak session is active for the test user.*
Test Sequence	

Step	*Type*	*Description*	*Result*
1	*Routing*	*Access a specific tool (e.g., BOMS) via the AccesS Suite URL. Verify Nginx correctly routes the request to the tool's dedicated server.*	*Successful*
2	*Communication*	*This test sequence evaluates the interaction between the AccesS Suite applications and the database. It includes checking whether queries, updates, and transactions initiated by the suite are correctly processed by the database, and that responses from the database are accurately reflected in the suite.*	*Successful*

Input Data	*User-facing workflows.*

Results	2026-01-19 10:00:01 | INFO | Test Started: AccesS Suite Routing & Communication Validation 2026-01-19 10:00:02 | INFO | Test Environment: Internal 2026-01-19 10:00:02 | INFO | Test Type: Functional / Integration 2026-01-19 10:01:05 | STEP 1 | Routing - Accessing a specific tool (e.g., BOMS) via the Hub URL 2026-01-19 10:01:06 | RESULT | Nginx correctly routed the request to the tool's dedicated server 2026-01-19 10:01:06 | STATUS | PASS 2026-01-19 10:02:10 | STEP 2 | Communication - Evaluating interaction between AccesS Suite and the database 2026-01-19 10:02:12 | RESULT | Queries, updates, and transactions processed correctly 2026-01-19 10:02:12 | RESULT | Responses accurately reflected in the suite 2026-01-19 10:02:12 | STATUS | PASS 2026-01-19 10:03:00 | INFO | Test Completed: All steps executed successfully 2026-01-19 10:03:00 | STATUS | OVERALL PASS

Table 5. AccesS Suite System Testing

Test Title	*AccesS Suite System Testing*
Identifier	*AS-DIST-SYS-01*
Test Type	*System Test*
AccesS Suite components	*All Components*
Tester(s)	*UNIS, CERTH, IsZEB*
Responsible	*UNIS, CERTH, IsZEB*
Test Purpose	*Validate that a Facility Manager can navigate the entire ecosystem, even though services are physically on different servers.*
Configuration	*HW/SW and OS configuration* • *OS: UBUNTU 22.04.5* • *Authentication Service: Keycloak* • *Database: TimescaleDB* • *Programming Language: Python, Node.js, React.js* • *GPU: RTX 4080*
Pre-test conditions	*All servers healthy;*
Test Sequence	

Step	***Type***	***Description***	***Result***
1	*E2E*	*Perform a full workflow: Login -> Navigation -> Logout*	*Successful*

	2	*Resilience*	*Simulate a failure on one Tool server; check if the rest of the Suite stays up*	*Successful*
Input Data	*User-facing workflows.*			
Results	2026-01-19 10:10:00 | INFO | Test Started: AccesS Suite E2E & Resilience Validation 2026-01-19 10:10:01 | INFO | Test Environment: Internal 2026-01-19 10:10:01 | INFO | Test Type: End-to-End / Resilience 2026-01-19 10:11:05 | STEP 1 | E2E - Perform a full workflow: Login -> Navigation -> Logout 2026-01-19 10:11:07 | RESULT | Login successful 2026-01-19 10:11:10 | RESULT | Navigation to target services completed successfully 2026-01-19 10:11:12 | RESULT | Logout executed successfully 2026-01-19 10:11:12 | STATUS | PASS 2026-01-19 10:12:15 | STEP 2 | Resilience - Simulate a failure on one Tool server 2026-01-19 10:12:18 | RESULT | Tool server failure simulated 2026-01-19 10:12:20 | RESULT | Rest of the AccesS Suite remained operational 2026-01-19 10:12:20 | STATUS | PASS 2026-01-19 10:13:00 | INFO | Test Completed: All steps executed successfully 2026-01-19 10:13:00 | STATUS | OVERALL PASS			

Energy Efficient & Sustainable Design Tool

Table 6. EESDT integration testing

Test Title	*EESDT Integration Testing*			
Identifier	*EESDT-INT-001*			
Test Type	*Integration Test*			
AccesS Suite components	• *EESDT Components* • *Keycloak authentication service*			
Tester(s)	*QUE, IsZEB*			
Responsible	*QUE, IsZEB*			
Test Purpose	*Validate that only authenticated users with valid tokens can access the tool and its IFC files, and that users can only view data and buildings they are authorized to access.*			
Configuration				
Pre-test conditions	• *Keycloak service running* • *EESDT registered as a client in Keycloack* • *AccesS Suite operational and accessible* • *EESDT Dashboard operational with pilot data*			
Test Sequence	**Step**	**Type**	**Description**	**Results**
	1	*Integration*	*Valid Token Access – Verify that a user with a valid access token can successfully access the EESDT functionality.*	*Successful*
	2	*Integration*	*Invalid or Expired Token – Ensure that requests with missing, expired, or tampered tokens are denied with proper error responses (HTTP 401).*	*Successful*
	3	*Integration*	*Role-Based Access – Check that users can only access features allowed by their roles and that unauthorized role attempts are blocked (HTTP 403).*	*Successful*
	4	*Integration*	*IFC File Access Control – Confirm that users can retrieve IFC files only for buildings they have permission to access, and unauthorized building requests are denied.*	*Successful*

	5	Integration	IFC File Integrity and Availability – Validate that retrieved IFC files are complete, uncorrupted, and properly processed, and that missing or inaccessible files are handled gracefully.	*Successful*
Input Data	• *User credentials (Username / Password)* • *BIM File in the database*			
Results	The integration tests confirmed that the EESDT reliably enforces authentication and authorization. Users with valid access tokens were able to access the tool and retrieve IFC files for buildings they are permitted to view, while requests with missing, expired, or invalid tokens were consistently denied. Role-based restrictions were correctly applied, preventing unauthorized access to restricted features. IFC files were retrieved intact, processed without errors, and appropriately handled when unavailable or corrupted. Overall, the tests demonstrate that the EESDT securely manages user access and maintains data integrity in accordance with assigned permissions.			

Energy Comfortness Tool

Table 7. Energy Comfortness Tool Unit Test

Test Title	**ECT – Personalized Comfort Calculation Using Different Occupant Profiles**			
Identifier	ECT-UNIS-FUNC-001			
Test Type	*Functional*			
AccesS Suite components	*Energy Comfortness Tool (v1 or v2*			
Tester(s)	*UNIS/ CERTH*			
Responsible	*UNIS*			
Test Purpose	*Verify that changing occupant profile (e.g. Adult-M vs Elderly-F with visual impairment) correctly affects thermal/visual comfort classes due to adjusted met/clo/visual models.*			
Configuration	*Configuration needed per component, concerning:* *CT v1 or v2* *Dataset with thermal + visual parameters*			
Pre-test conditions	*At least two profiles available (e.g. Adult-M, Elderly-F-visual)*			
Test Sequence	**Step**	**Type**	**Description**	**Result**
	1	Select	Choose profile Adult-M → run thermal & visual tabs	Baseline PMV/PPD and visual score
	2	Change	Switch to Elderly-F (higher clo, visual impairment)	Higher PPD (colder sensation), lower visual score
	3	Validate	Compare overall comfortness class	Possible downgrade (e.g. B → C) due to personalization

Input Data	*Data type and format* Same sensor time-series, different profile_id			
Results	Thermal & Visual tabs side-by-side for both profiles + overall comfortness comparison (different results with different profiles)			

Table 8. Energy Comfortness Tool Integration & Security Test

Test Title	ECT Integrated Version – Successful Authentication via AccesS Suite Keycloak and Access to Dashboard			
Identifier	*ECT-UNIS-INT-001*			
Test Type	*Integration / Security*			
AccesS Suite components	*Energy Comfortness Tool (integrated), AccesS Suite Authentication (Keycloak), AccesS Suite API Gateway*			
Tester(s)	*UNIS / CERTH*			
Responsible	*UNIS*			
Test Purpose	*Verify that the integrated ECT redirects unauthenticated users to the central AccesS Keycloak login, performs successful authentication, returns the user to the ECT dashboard, displays the correct user identity, and grants access only to authorized roles.*			
Configuration	*Configuration needed per component, concerning:* • *ECT (integrated version) deployed in containerized environment (Docker as per D4.5)* • *Connected to AccesS Suite Keycloak instance* • *Test user account pre-registered in Keycloak with role allowing ECT access* • *Browser: IE/Chrome / Firefox (latest)* • *Network: internal project VPN or public internet with correct CORS settings*			
Pre-test conditions	• *ECT service running and reachable (https://ect.city4life.eu/)* • *Keycloak server operational and configured with AccesS client* • *At least one pilot building/space with data already ingested in AccesS Data Hub* • *Tester has valid Keycloak credentials (username/password)*			
Test Sequence	**Step**	**Type**	**Description**	**Result**

	1	*Navigation*	*Open ECT landing page in browser without prior login*	*Redirect to AccesS Suite login page*
	2	*Authentication*	*Enter valid credentials and submit*	*Successful login → redirect back to ECT*
		UI Verification	*Check initial ECT dashboard screen after redirect*	*User identity displayed*
		Functional	*Navigate to Thermal comfort tab (Choose Profile /Pilot/Space by left side bar)*	*Building/space metadata loaded automatically real data sensor from AccesS Suite*
		Negative	*Log out and re-access direct ECT URL*	*Redirect back to Access Suite platform*
Input Data	*Data type and format* • *Valid Keycloak test user credentials for ECT* • *No manual data upload (data must come from Access Suite Platform)*			
Results	*Successful authentication & viewed related info on the Thermal comfort tab (auto-loaded pilot data (Droom je Thuis example)*			

Table 9. Energy Comfortness Tool Integration Test

Test Title	ECT Integrated – Automated Ingestion of Pilot Data from AccesS Suite Data Hub
Identifier	ECT-UNIS-INT-002
Test Type	*Integration / Functional*
AccesS Suite components	*Energy Comfortness Tool (integrated), AccesS Data Hub, AccesS API Gateway*

Tester(s)	*UNIS*
Responsible	*UNIS*
Test Purpose	*Confirm that ECT (v2) automatically retrieves sensor data, building metadata, and occupant profiles via API without manual CSV/IFC upload, and correctly populates the dashboard.*
Configuration	*Configuration needed per component, concerning:* • *ECT v2 container running* • *AccesS Suite Data Hub populated with Droom je Thuis pilot data (Jul–Sep 2025)* • *API credentials / token configured in ECT*
Pre-test conditions	• *Pilot data (temperature,P.M.2.5, CO_2, etc.) available in Data Hub for space "BHA-01"* • *Occupant profile "Adult-M" exists in Suite* • *No local CSV/IFC files present in ECT container*

Test Sequence

Step	Type	Description	Result
1	*Trigger*	*Log in via Suite API and open Thermal tab*	*No "upload data" prompt appears*
2	*Data Fetch API*	*Select building = Droom je Thuis, space = BHA-01, period Jul-Sep 2025 within related APO*	*Data automatically loaded; UI Chart graphs show values*
3	*Validation*	*Check displayed metadata (building_id, space_id, profile)*	*Matches Access Suite records*
4	*Negative*	*Select non-existing space ID*	*API Response message "No data available for this space"*

Input Data	*Data type and format* • *API-driven (no manual input files)*
Results	• *Thermal tab with time-series and comfort class distribution; metadata panel aligned with Access Suite API message (JSON payload)*

Table 10. Energy Comfortness Tool System Test

Test Title	**ECT – Consistency of Overall Comfortness Calculation between Standalone (v1) and Integrated (v2)**
Identifier	ECT-UNIS-COMP-001
Test Type	*Functional / Regression*
AccesS Suite components	*Energy Comfortness Tool v1 (standalone), Energy Comfortness Tool v2 (integrated)*
Tester(s)	*UNIS*
Responsible	*UNIS*
Test Purpose	*Verify that the same input dataset produces identical comfort class distributions and overall comfortness score in both versions (proving common core invariance – D4.2 §3.2 & D4.5 §3.2).*
Configuration	*Configuration needed per component, concerning:* • *Standalone ECT v1 running locally (Docker)* • *Integrated ECT v2 connected to Suite* • *Identical dataset (exported from Suite or CERTH Smart Home example)*
Pre-test conditions	• *Same occupant profile selected (e.g. Adult-M)* • *Same time period and space* • *Same weighting factors for overall comfortness*
Test Sequence	

Step	Type	Description	Result
1	*Data Prep*	*Export pilot dataset from Suite (or use CERTH sample CSV)*	*CSV ready*
2	*Standalone*	*Import CSV to ECT v1 → run simulation → record overall comfortness class*	*e.g. Class C, 42% A/B, PPD avg 18.7*
3	*Integrated*	*Ingest same data via Suite → run in ECT v2 → record results*	*Identical class distribution & overall class*
4	*Comparison*	*Compare pie charts and numerical values (PMV/PPD, domain classes*	*Zero deviation in computed comfort indices*

Input Data	*Data type and format* • *Identical time-series CSV (temperature, RH, CO_2, etc.) + occupant profile*
Results	• *Same results from both versions*

ErgoFlow

Table 11. ErgoFlow Ergonomics, Comfort and Accessibility Evaluation and Library Validation Test

Test Title	ErgoFlow Ergonomics, Comfort and Accessibility Evaluation and Library Validation Test
Identifier	*Unique identifier of the test*
Test Type	*E.g.: Integration, Unit, Security, System*
AccesS Suite components	Ergonomics and Comfort Design of Products
Tester(s)	*CERTH, BHA*
Responsible	CETEM
Test Purpose	to validate the correct operation, reliability, and usability of the ErgoFlow Evaluation Tool and ErgoFlow Library in assessing the ergonomics, comfort, and accessibility of furniture products. Specifically, the test aims to verify that: -Furniture 3D models can be uploaded and visualized correctly. -Ergonomic and accessibility parameters can be manually introduced according to predefined criteria. -The system correctly applies weighted scoring based on internationally recognized standards. -The ergonomic score threshold (≥65%) is correctly calculated and applied. -Evaluation reports are generated accurately. -Compliant models can be uploaded to and retrieved from the ErgoFlow Library.
Configuration	**HW / SW and OS Configuration** -Web-based system accessed through a modern web browser. -ErgoFlow implemented as a modular monolithic web application based on Next.js 15, React 19, Node.js runtime, MongoDB (data persistence), Cloudinary (file storage) and Keycloak (authentication and RBAC) **Network Configuration** -Internet connectivity required. -Secure HTTPS communication. -REST API communication between frontend and backend. -Authentication through the AccesS Suite single sign-on mechanism (Keycloak).
Pre-test conditions	-ErgoFlow Evaluation Tool is deployed and accessible via web interface. -Authentication through the AccesS Suite is operational. -MongoDB database and Cloudinary storage are available and correctly configured.

	-At least one test user account with appropriate role permissions exists. -Test 3D models in .glb format (≤10 MB) and reference images (.jpg, .png, .jpeg) are available.
Test Sequence	

Step	Type	Description	Result
1	Network	Access ErgoFlow via browser and authenticate through AccesS SSO	*Successful*
2	Functional	Create a new project and select a furniture type	*Successful*
3	Functional	Upload reference image and 3D .glb model	*Successful*
4	Functional	Visualize and interact with 3D model	*Successful*
5	Functional	Manually input ergonomic and accessibility parameters	*Successful*
6	Functional	Execute ergonomic evaluation	*Successful*
7	Functional	Review ergonomic score and parameter-level feedback	*Successful*
8	Functional	Download PDF evaluation report	*Successful*
9	Integration	Upload compliant model (≥65%) to ErgoFlow Library	*Successful*
10	Integration	Retrieve model and evaluation data from Library	*Successful*

Input Data	**3D Models:** -Format: .glb -Max size: 10 MB **Images:** -Formats: .jpg, .png, .jpeg -Max size: 10 MB **Manual Parameters:** -Numeric values (e.g. seat height, depth, clearance dimensions) -Boolean values (e.g. adjustable height, non-slip surface) -Categorical values (e.g. furniture type, surface shape)
Results	The ErgoFlow Evaluation Tool and the ErgoFlow Library were assessed through internal peer reviews performed by CERTH and BRUCO within the framework of T4.3, prior to the submission of Deliverable D4.6. The detailed findings, comments, and recommendations from both reviewers are documented in their respective peer review reports. The review conducted by CERTH resulted in a full acceptance of the deliverable, with only minor suggestions and no required changes to the core methodology, while the review performed by BRUCO resulted in an acceptance with reservations, identifying several practical limitations and providing recommendations for improvement. No major changes to the deliverable were required in either case.

VUMs/VHs

Table 12. VUMs/VHs GUI Editor System and Integration Test Description

Test Title	*System and Integration Testing of GUI Editor for VUMs and VHs*
Identifier	*FORTH-VUMVH-SYS-01*
Test Type	*System / Integration*
AccesS Suite components	*Virtual User Models and VUM-based Virtual Humans*
Tester(s)	*FORTH*
Responsible	*FORTH*
Test Purpose	*Validate end-to-end operation of the GUI editor for digital authoring of VUMs and generation of corresponding Virtual Humans (VHs).* *The test evaluates:* • *VUM creation/editing/deletion workflow* • *Application of default metric values and disability effects* • *Integration between GUI and backend REST API services* • *VH generation from a selected VUM and mapping of parameters to 3D model* • *Export/download of the generated VH 3D model (.glb)*
Configuration	• ***Web Frontend:*** *Deployed on a web server and accessible externally.* • ***REST API:*** *Deployed as a Docker container and exposed externally through controlled endpoints.* • ***Database****: MongoDB used to store configuration data and auxiliary metadata.* • ***Ontology Storage****: Neo4j graph database used to store and query the formal VUM ontology representing aging individuals and people with disabilities.* • ***Object Storage****: MinIO used for storing generated assets such as VH 3D files.* • ***Authentication & Authorization****: Keycloak used for user authentication and Single Sign-On (SSO) across the Access Suite.*

	• ***Deployment & Network:*** *Docker-based deployment with private internal network; only Web Frontend and REST API are exposed externally, while rest of the services remain isolated.* • ***Network configuration:*** *Secure communication between containers; HTTPS enabled for external access where applicable.*
Pre-test conditions	• *Web Frontend is deployed and accessible via its public URL.* • *REST API is deployed and reachable from the frontend.* • *Neo4j database is initialized with the VUM ontology schema.* • *MongoDB instance is running* • *Keycloak is deployed, configured, and integrated with the frontend and REST API.* • *Docker containers are running within a private network with correct service-to-service connectivity.* • *Test user accounts are available.*

Test Sequence

Step	Type	Description	Result
1	*Functional*	*Authenticate via Keycloak SSO and access the VUM/VH GUI Editor.*	*Successful*
2	*Functional*	*Create a new VUM by completing general information (Step 1), selecting disability types and attributes (Step 2), and fine-tuning selected metrics (Step 3); save the VUM.*	*Successful*
3	*Integration*	*Create a new VH by selecting the previously created VUM.*	*Successful*
4	*Integration*	*Select a base model (VH Step 2) and verify parameter-driven changes (e.g., height/weight/age, mobility aid/pose).*	*Successful*
5	*Functional*	*Fine-tune VH appearance (VH Step 3: clothing, hair, accessories) and finalize.*	*Successful*
6	*Functional*	*Export/download VH as a .glb file.*	*Successful*
7	*Integration*	*Edit the VUM and regenerate/update the VH; verify consistency of updated parameters.*	*Successful*
8	*Functional*	*Delete test VUM/VH and confirm removal from UI and backend.*	*Successful*

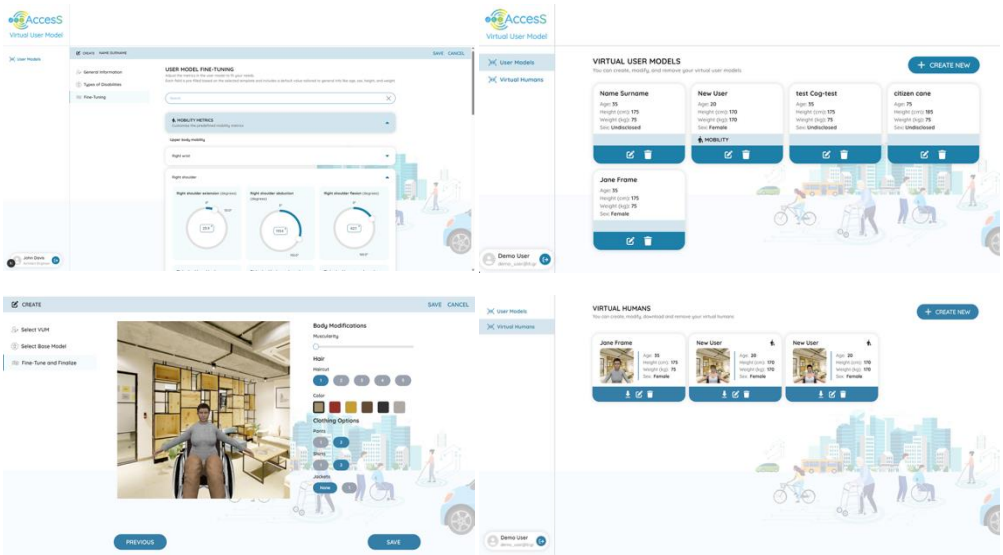
Input Data	• *User demographic data (age, sex, height, weight).* • *Disability attributes (type, specification, severity, placement).* • *VH stylistic parameters (hair/clothing/accessories), selected mobility aid type.*
Results	 • *VUMs are created/edited/deleted correctly and persist as expected.* • *Default values and disability effects are applied correctly to the VUM metrics.* • *VHs reflect the selected VUM parameters and update correctly after VUM edits.* • *Exported .glb file is generated successfully and is usable in the Access Suite 3D Scenarios Simulator.*

Table 13. VUMs/VHs GUI Editor Security Test Description

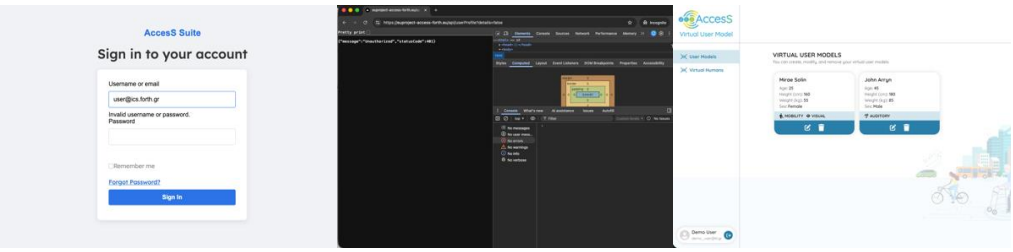
Test Title	*Security Testing of GUI Editor for VUMs and VHs*
Identifier	*FORTH-VUMVH-SEC-01*
Test Type	*Security*
AccesS Suite components	*Virtual User Models and VUM-based Virtual Humans*
Tester(s)	*FORTH*
Responsible	*FORTH*
Test Purpose	*Assess robustness against unauthorized access, data leakage, and malicious usage.*

<table>
<tr><td>Configuration</td><td colspan="4">

Web Frontend: Deployed on a web server and accessible externally.
REST API: Deployed as a Docker container and exposed externally through controlled endpoints.
Database: MongoDB used to store configuration data and auxiliary metadata.
Ontology Storage: Neo4j graph database used to store and query the formal VUM ontology representing aging individuals and people with disabilities.
Object Storage: MinIO used for storing generated assets such as VH 3D files.
Authentication & Authorization: Keycloak used for user authentication and Single Sign-On (SSO) across the Access Suite.
Deployment & Network: Docker-based deployment with private internal network; only Web Frontend and REST API are exposed externally, while rest of the services remain isolated.
Network configuration: Secure communication between containers; HTTPS enabled for external access where applicable.

</td></tr>
<tr><td>Pre-test conditions</td><td colspan="4">

User roles configured,
Test user accounts available
Secured API endpoints

</td></tr>
<tr><td>Test Sequence</td><td>Step</td><td>Type</td><td>Description</td><td>Result</td></tr>
<tr><td></td><td>1</td><td>Functional</td><td>Invalid login attempt</td><td>Access denied</td></tr>
<tr><td></td><td>2</td><td>Functional</td><td>Unauthorized GUI access</td><td>Access denied</td></tr>
<tr><td></td><td>3</td><td>Functional</td><td>Cross-user VUM access</td><td>Access denied</td></tr>
<tr><td></td><td>4</td><td>Functional</td><td>Unauthorized API call</td><td>Access denied</td></tr>
<tr><td></td><td>5</td><td>Integration</td><td>Malformed input submission</td><td>Safely rejected</td></tr>
<tr><td>Input Data</td><td colspan="4">Authorized/unauthorized requests, malformed JSON payloads</td></tr>
<tr><td>Results</td><td colspan="4"> 

Unauthorized access blocked, data confidentiality preserved</td></tr>
</table>

3D Simulation Environments

Table 14. 3D Simulation Environments Unit Test

Test Title	**3D Simulator Validation**			
Identifier	3D_Simulator_Dynamic_Analysis			
Test Type	Unit test			
AccesS Suite components	VUMs, SANTO, Simulation tool			
Tester(s)	CERTH			
Responsible	CERTH			
Test Purpose	Test of the dynamic simulation validity			
Configuration	-			
Pre-test conditions	Integration between SANTO and VUMs with 3D Simulator environment			
Test Sequence	**Step**	**Type**	**Description**	**Result**
	1	*Simulation integrity*	*Dynamic simulation results align with data from the literature regarding forces applied on joints*	*Successful*
Input Data	JSON VUMs & JSON SANTO			
Results	The simulation dynamic analysis output matches the data found in the literature regarding forces applied on knee joints during normal walking as well as ascending and descending stairs tasks.			

Table 15. 3D Simulation Environments Integration Test

Test Title	Test in Simulation Application integration with SANTO & VUMs API			
Identifier	Unity_Santo_VUM			
Test Type	Integration			
AccesS Suite components	VUMs, SANTO, Simulation tool			
Tester(s)	CERTH			
Responsible	CERTH/ FORTH			
Test Purpose	Test the interoperability between SANTO and VUM editor with the 3D Simulator			
Configuration	-			
Pre-test conditions	IFC Model edited in SANTO application			
Test Sequence	**Step**	**Type**	**Description**	**Result**

	1	*Connectivity*	*Communication with FORTH Backend service*	*Successful*
	2	*Data integrity*	*JSON format as expected*	*Successful*
Input Data	JSON VUMs & JSON SANTO			
Results	-			

BOMS TESTS

Table 16. BOMS Unit Testing

Test Title	*BOMS Unit Testing*
Identifier	*BOMS-UNIT-001*
Test Type	*Unit Test*
AccesS Suite components	*BOMS*
Tester(s)	*CERTH, IsZEB*
Responsible	*CERTH, IsZEB*
Test Purpose	*Verify that the individual functions of the BOMS component work correctly in isolation using valid input data (forecast engine, decision support tool, BOMS Dashboard)*

Configuration	*Configuration needed per component, concerning:*
	• *HW/SW and OS configuration*
	  ○ *OS: UBUNTU 22.04.5*
	  ○ *Programming Language: Python, Node.js, React.js*

Pre-test conditions	• *BOMS source code available* • *All required libraries installed* • *Unit test framework configured*
Test Sequence	

Step	Type	Description	Result
1	*Functional*	*Execute unit test (forecast engine, decision support tool, BOMS Dashboard)*	*Successful*
2	*Functional*	*Verify individual function outputs*	*Successful*

Input Data	*Function parameters* *Format: JSON/numeric values*
Results	

```
2026-01-19 10:20:00 | INFO  | Test Started: BOMS Functional Unit Tests
2026-01-19 10:20:01 | INFO  | Test Environment: Internal
2026-01-19 10:20:01 | INFO  | Test Type: Functional / Unit Tests
2026-01-19 10:20:01 | INFO  | Module Under Test: BOMS Dashboard, Forecast Engine, Decision Support Tool

2026-01-19 10:21:05 | STEP 1 | Functional - Execute unit test on BOMS Dashboard, Forecast Engine, and Decision Support Tool
2026-01-19 10:21:08 | RESULT | Unit tests executed successfully
2026-01-19 10:21:08 | STATUS | PASS

2026-01-19 10:22:10 | STEP 2 | Functional - Verify individual function outputs in BOMS
2026-01-19 10:22:12 | RESULT | All individual function outputs verified successfully
2026-01-19 10:22:12 | STATUS | PASS

2026-01-19 10:23:00 | INFO  | Test Completed: All BOMS functional unit test steps executed successfully
2026-01-19 10:23:00 | STATUS | OVERALL PASS
```

Table 17. BOMS Integration Testing

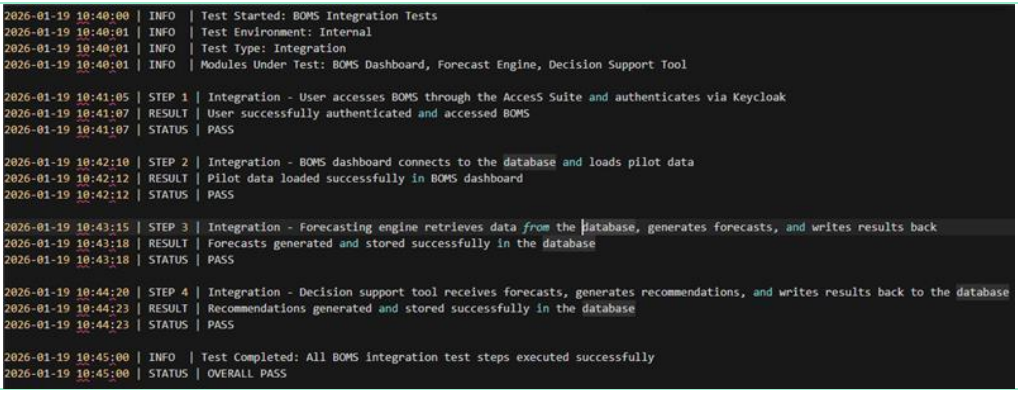
Test Title	*BOMS Integration Testing*
Identifier	*BOMS-INT-001*
Test Type	*Integration Test*

AccesS Suite components	• *BOMS Components:*
	  ○ *BOMS Dashboard*
	  ○ *Forecasting engine*
	  ○ *Decision support tool*
	• *Keycloak authentication*

Tester(s)	*CERTH, IsZEB*
Responsible	*CERTH, IsZEB*
Test Purpose	*Validate that BOMS is correctly integrated with the Access Suite via Keycloak and that all BOMS ecosystem components work together and connect to the database, including monitoring pilot data, generating forecasts, and providing recommendations.*

Configuration	*Configuration needed per component, concerning:*
	• *HW/SW and OS configuration*
	  ○ *OS: UBUNTU 22.04.5*
	  ○ *Authentication Service: Keycloak*
	  ○ *Database: TimescaleDB*
	  ○ *Programming Language: Python, Node.js, React.js*
	  ○ *GPU: RTX 4080*
	• *Network configuration:*
	  ○ *HTTPS enabled for BOMS*

Pre-test conditions	• *Keycloak service running* • *BOMS registered as a client in Keycloack* • *AccesS Suite operational and accessible* • *BOMS Dashboard operational with pilot data* • *Forecasting and recommendation modules running*

Test Sequence	Step	Type	Description	Results
	1	*Functional*	*User accesses BOMS through the AccesS Suite and authenticate via Keycloack*	*Successful*
	2	*Functional*	*BOMS dashboard connects to the database and loads pilot data*	*Successful*
	3	*Functional*	*Forecasting engine retrieves data from the database, generate forecasts, and writes back to the database*	*Successful*
	4	*Functional*	*Decision support tool receives forecasts, generates recommendations, and writes results back to the database.*	*Successful*
Input Data	• *User credentials (Username / Password)* • *Pilot/test data in the database*			
Results				

```
2026-01-19 10:40:00 | INFO  | Test Started: BOMS Integration Tests
2026-01-19 10:40:01 | INFO  | Test Environment: Internal
2026-01-19 10:40:01 | INFO  | Test Type: Integration
2026-01-19 10:40:01 | INFO  | Modules Under Test: BOMS Dashboard, Forecast Engine, Decision Support Tool

2026-01-19 10:41:05 | STEP 1 | Integration - User accesses BOMS through the AccesS Suite and authenticates via Keycloak
2026-01-19 10:41:07 | RESULT | User successfully authenticated and accessed BOMS
2026-01-19 10:41:07 | STATUS | PASS

2026-01-19 10:42:10 | STEP 2 | Integration - BOMS dashboard connects to the database and loads pilot data
2026-01-19 10:42:12 | RESULT | Pilot data loaded successfully in BOMS dashboard
2026-01-19 10:42:12 | STATUS | PASS

2026-01-19 10:43:15 | STEP 3 | Integration - Forecasting engine retrieves data from the database, generates forecasts, and writes results back
2026-01-19 10:43:18 | RESULT | Forecasts generated and stored successfully in the database
2026-01-19 10:43:18 | STATUS | PASS

2026-01-19 10:44:20 | STEP 4 | Integration - Decision support tool receives forecasts, generates recommendations, and writes results back to the database
2026-01-19 10:44:23 | RESULT | Recommendations generated and stored successfully in the database
2026-01-19 10:44:23 | STATUS | PASS

2026-01-19 10:45:00 | INFO  | Test Completed: All BOMS integration test steps executed successfully
2026-01-19 10:45:00 | STATUS | OVERALL PASS
```

Table 18. BOMS Security Testing

Test Title	*BOMS Security Testing*
Identifier	*BOMS-SEC-001*
Test Type	*Security Test*
AccesS Suite components	*BOMS Secure Test Environment*
Tester(s)	*CERTH , IsZEB*
Responsible	*CERTH , IsZEB*

Test Purpose	*Ensure that BOMS enforces secure user authentication and access through Keycloak and protects user sessions and data.*

Configuration	*Configuration needed per component, concerning:*
	• *HW/SW and OS configuration*
	  ○ *OS: UBUNTU 22.04.5*
	  ○ *Authentication Service: Keycloak*
	  ○ *Database: TimescaleDB*
	  ○ *Programming Language: Python, Node.js, React.js*
	  ○ *GPU: RTX 4080*
	• *Network configuration:*
	  ○ *HTTPS enabled for BOMS*

Pre-test conditions	• *Keycloak service running* • *Valid and invalid user accounts available* • *BOMS configured as a Keycloak client* • *Access roles defined*

Test Sequence

Step	Type	Description	Result
1	*Security*	*Attempt login with valid credentials*	*Successful*
2	*Security*	*Attempt login with invalid credentials*	*Successful*
3	*Security*	*Attempt access to BOMS without authentication*	*Successful*
4	*Security*	*Verify role-based access control*	*Successful*
5	*Security*	*Verify secure session handling (logout/ session expiration)*	*Successful*

Input Data	*User credentials (Username / Password)*

Results	2026-01-19 10:50:00 | INFO | Test Started: BOMS Security Tests 2026-01-19 10:50:01 | INFO | Test Environment: Internal 2026-01-19 10:50:01 | INFO | Test Type: Security 2026-01-19 10:50:01 | INFO | Modules Under Test: BOMS Dashboard, Authentication, Role Management 2026-01-19 10:51:05 | STEP 1 | Security - Attempt login with valid credentials 2026-01-19 10:51:07 | RESULT | User successfully logged in 2026-01-19 10:51:07 | STATUS | PASS 2026-01-19 10:52:10 | STEP 2 | Security - Attempt login with invalid credentials 2026-01-19 10:52:12 | RESULT | Login attempt rejected as expected 2026-01-19 10:52:12 | STATUS | PASS 2026-01-19 10:53:15 | STEP 3 | Security - Attempt access to BOMS without authentication 2026-01-19 10:53:18 | RESULT | Access denied as expected 2026-01-19 10:53:18 | STATUS | PASS 2026-01-19 10:54:20 | STEP 4 | Security - Verify role-based access control 2026-01-19 10:54:23 | RESULT | Users granted/denied access according to roles 2026-01-19 10:54:23 | STATUS | PASS 2026-01-19 10:55:25 | STEP 5 | Security - Verify secure session handling (logout/session expiration) 2026-01-19 10:55:28 | RESULT | Sessions terminated correctly upon logout or expiration 2026-01-19 10:55:28 | STATUS | PASS 2026-01-19 10:56:00 | INFO | Test Completed: All BOMS security test steps executed successfully 2026-01-19 10:56:00 | STATUS | OVERALL PASS

Table 19. BOMS System Workflow Testing – End-to-End Operation

Test Title	*BOMS System Workflow Testing – End-to-End Operation*
Identifier	*BOMS-SYS-001*
Test Type	*System Workflow Test*
AccesS Suite components	*BOMS*
Tester(s)	*CERTH, IsZEB*
Responsible	*CERTH, IsZEB*
Test Purpose	*Validate that a user can access BOMS through the Access Suite and complete normal operations, including monitoring pilot data, running forecasts, and receiving recommendations.*
Configuration	*Configuration needed per component, concerning:* • *HW/SW and OS configuration* ○ *OS: UBUNTU 22.04.5*

	○ *Authentication Service: Keycloak* ○ *Database: TimescaleDB* ○ *Programming Language: Python, Node.js, React.js* ○ *GPU: RTX 4080* • *Network configuration:* • *HTTPS enabled for BOMS*
Pre-test conditions	*Pre-test Conditions* • *Access Suite operational* • *Keycloak service running* • *Valid user account and role defined* • *BOMS deployed and accessible* • *Forecasting and recommendation modules running*
Test Sequence	

Step	Type	Description	Result
1	*Functional*	*User accesses the AccesS Suite*	*Successful*
2	*Security*	*User authenticates via Keycloak*	*Successful*
3	*Functional*	*User selects and launches BOMS from the AccesS Suite*	*Successful*
4	*Functional*	*BOMS loads with user-specific session and permissions*	*Successful*
5	*Functional*	*BOMS retrieves and displays pilot data from the database*	*Successful*
6	*Functional*	*BOMS run forecasts using the forecasting engine*	*Successful*
7	*Functional*	*BOMS run forecasts using the recommendation tool*	*Successful*
8	*Functional*	*User logs out from the system*	*Successful*

Input Data	*User credentials (Username / Password)* *Pilot/test data and UI inputs*

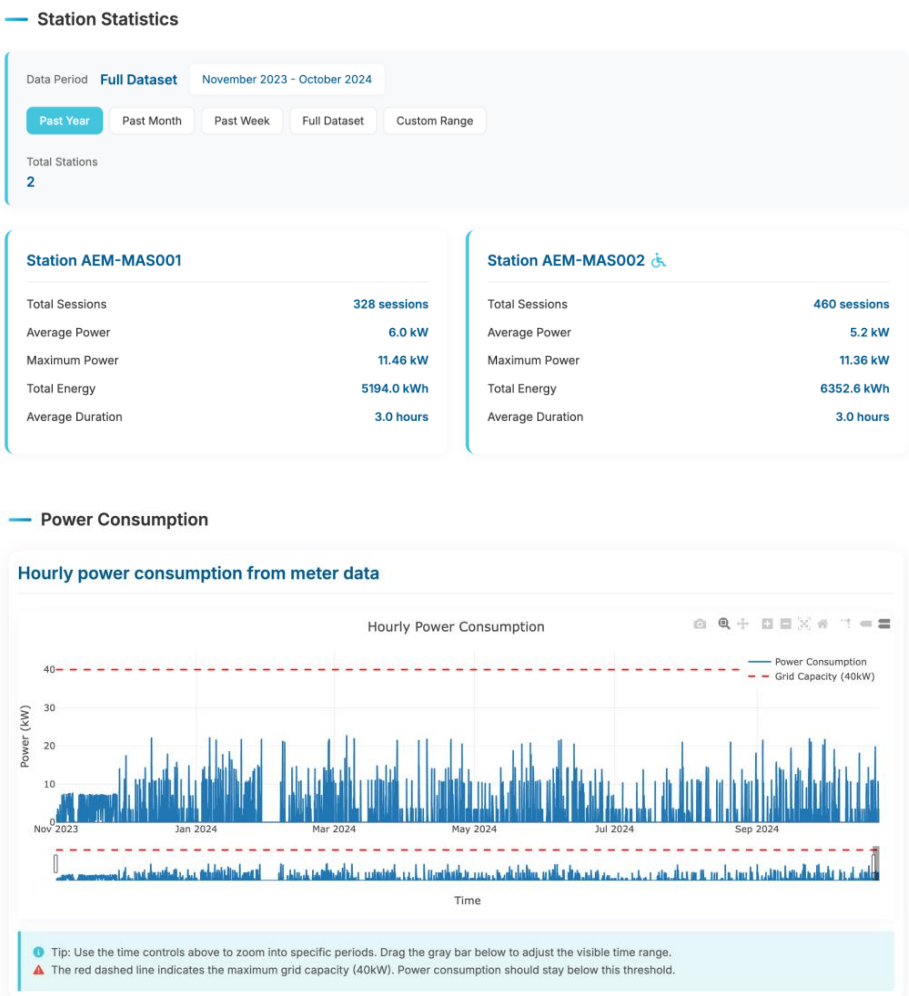
Results

```
2026-01-19 11:00:01 | INFO  | Test Environment: Internal
2026-01-19 11:00:01 | INFO  | Test Type: Workflow
2026-01-19 11:00:01 | INFO  | Modules Under Test: AccesS Suite, BOMS Dashboard, Forecast Engine, Recommendation Tool

2026-01-19 11:01:05 | STEP 1 | Functional - User accesses the AccesS Suite
2026-01-19 11:01:07 | RESULT | AccesS Suite accessed successfully
2026-01-19 11:01:07 | STATUS | PASS

2026-01-19 11:02:10 | STEP 2 | Security - User authenticates via Keycloak
2026-01-19 11:02:12 | RESULT | User authenticated successfully
2026-01-19 11:02:12 | STATUS | PASS

2026-01-19 11:03:15 | STEP 3 | Functional - User selects and launches BOMS from the AccesS Suite
2026-01-19 11:03:18 | RESULT | BOMS launched successfully
2026-01-19 11:03:18 | STATUS | PASS

2026-01-19 11:04:20 | STEP 4 | Functional - BOMS loads with user-specific session and permissions
2026-01-19 11:04:23 | RESULT | Session and permissions applied correctly
2026-01-19 11:04:23 | STATUS | PASS

2026-01-19 11:05:25 | STEP 5 | Functional - BOMS retrieves and displays pilot data from the database
2026-01-19 11:05:28 | RESULT | Pilot data loaded and displayed successfully
2026-01-19 11:05:28 | STATUS | PASS

2026-01-19 11:06:30 | STEP 6 | Functional - BOMS runs forecasts using the forecasting engine
2026-01-19 11:06:33 | RESULT | Forecasts generated successfully
2026-01-19 11:06:33 | STATUS | PASS

2026-01-19 11:07:35 | STEP 7 | Functional - BOMS runs forecasts using the recommendation tool
2026-01-19 11:07:38 | RESULT | Recommendations generated successfully
2026-01-19 11:07:38 | STATUS | PASS

2026-01-19 11:08:40 | STEP 8 | Functional - User logs out from the system
2026-01-19 11:08:43 | RESULT | Logout executed successfully
2026-01-19 11:08:43 | STATUS | PASS

2026-01-19 11:09:00 | INFO  | Test Completed: All BOMS workflow test steps executed successfully
2026-01-19 11:09:00 | STATUS | OVERALL PASS
```

EV Charging Optimizer

Table 20. Integration and functional testing of the EV Charging Optimizer

Test Title	*Integration and functional testing of the EV Charging Optimizer*			
Identifier	*AccesS_EV Optimizer*			
Test Type	*System, Integration, Functional*			
AccesS Suite components	*AccesS Suite, EV Charging Optimizer*			
Tester(s)	*HIVE*			
Responsible	*HIVE*			
Test Purpose	• *Ensure the EV Charging Optimizer is accessible through the AccesS Suite* • *Access works via the Keycloak role-based authentication system* • *EV Charging Optimizer runs daily and data is updated* • *Data is correctly received as input from the field integration*			
Configuration	*Configuration needed per component, concerning:* • *HW/SW: back-end running on Python, front-end HTML/JavaScript and Plotly.js for interactive visualizations, authentication Keycloak* • *Network configuration: secure HTTPS to AccesS Suite, API connectivity for field data*			
Pre-test conditions	• *Charging station data available via APIs* • *Accessibility criteria defined as input* • *Keycloak roles defined* • *Keycloak authentication service operational*			
Test Sequence	*Step*	Type	Description	Result
	1	*Network*	Established integration between EV Charging Optimizer and AccesS Suite	Successful
	2	Functional	Keycloak authentication works correctly	Successful
	3	*Functional*	Charging station data is available	Successful

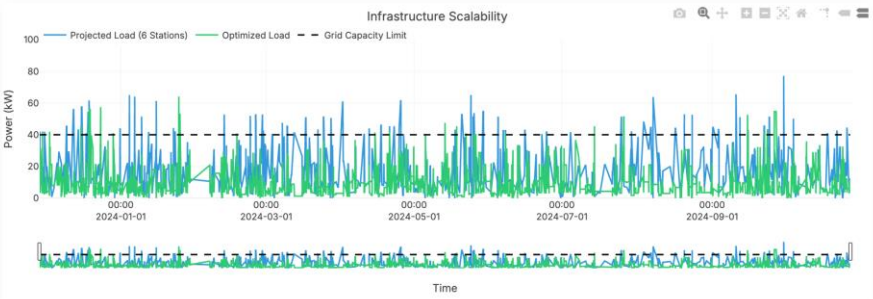
	4	Functional	Python code runs correctly	Successful
	5	UI	EV driver dashboard data visualization	Successful
	6	UI	CPO dashboard data visualization	Successful
Input Data	***Charging session data*** • *Format: JSON / CSV* • *Fields: timestamp, charging power, session duration, station ID*			
Results				

Infrastructure Scalability

Projected Load with 6 Charging Stations

This simulation shows the projected power load when scaling from 2 to 6 charging stations.

- Projected load with 6 stations
- Optimized load with intelligent scheduling / load shifting
- Grid capacity limit (40kW)



Note: Charging session shifting helps minimize peak loads, but there are still moments when grid capacity is exceeded. Combining load shifting with software-based power modulation helps avoid these peaks and ensures a more balanced energy distribution.

Accessibility Assessment Tool (AAT)

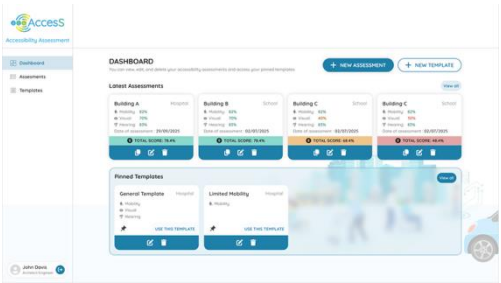
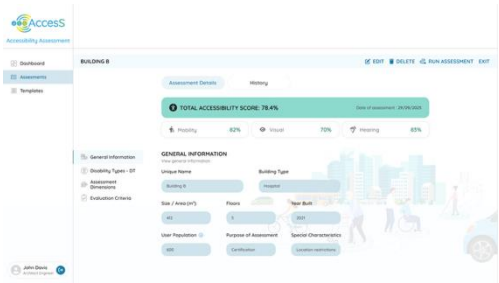
Table 21. AAT System and Integration Test

Test Title	*System and Integration Testing of the Large-Scale Accessibility Assessment Tool*
Identifier	*FORTH-AAT-SYS-01*
Test Type	*System / Integration*
AccesS Suite components	*Large-Scale Accessibility Assessment Tool*
Tester(s)	*FORTH*
Responsible	*FORTH*
Test Purpose	*Verify the end-to-end workflow for assessing the design and operation of building and infrastructure according to the developed Accessibility Assessment Scheme.* *The test evaluates:* • *Selection of disability types and assessment dimensions/evaluation criteria* • *Calculation of accessibility scores (e.g. total impact score, overall building score)* • *Integrity and traceability of results (repeatability under same inputs)*
Configuration	• ***Web Frontend:*** *Deployed on a web server and accessible externally.* • ***REST API:*** *Deployed as a Docker container and exposed externally through controlled endpoints.* • ***Database:*** *MongoDB used to store the developed Accessibility Assessment Scheme and related configuration data.* • ***Object Storage:*** *MinIO used as an object storage service for assessment reports and generated files.* • ***Authentication:*** *Keycloak embedded for user authentication and Single Sign-On (SSO) across the Access Suite.* • ***Deployment & Network:*** *Docker-based deployment with private internal network; only Web Frontend and REST API are exposed externally, while MongoDB, MinIO, and internal services remain isolated.* • ***Network configuration:*** *Secure communication between containers; HTTPS enabled for external access where applicable.*

Pre-test conditions	• *Web Frontend is deployed and accessible via its public URL.* • *REST API is deployed and reachable from the frontend.* • *MongoDB instance is running and populated with the Accessibility Assessment Scheme.* • *MinIO service is running and accessible from the REST API.* • *Keycloak is deployed, configured, and integrated with the frontend and REST API.* • *Docker containers are running within a private network with correct service-to-service connectivity.* • *Test building/environment models and assessment data are available.*

Test Sequence

Step	Type	Description	Result
1	*Functional*	*Access the AAT Web Frontend and authenticate via Keycloak SSO.*	*Successful*
2	*Integration*	*Frontend retrieves Assessment Scheme data from REST API.*	*In progress*
3	*Functional*	*User selects the associated weights to reflect importance of disability types, assessment dimensions, and evaluation criteria.*	*In progress*
4	*Integration*	*Frontend triggers score computation; REST API applies the mathematical model and returns total impact score and overall building score.*	*In progress*
5	*Integration*	*REST API stores generated assessment report in MinIO object storage.*	*In progress*
6	*Functional*	*Frontend retrieves and displays results and provides access to the generated report.*	*In progress*
7	*Integration*	*Repeat assessment with identical inputs to verify consistency of scores and stored outputs.*	*In progress*

Input Data	• *Building associated metadata.* • *Accessibility Assessment Scheme data and scoring mathematical model stored in MongoDB.* • *User-selected associated weights of disability types, assessment dimensions, and evaluation criteria.* • *Scoring parameters used by the mathematical model.*
Results	 

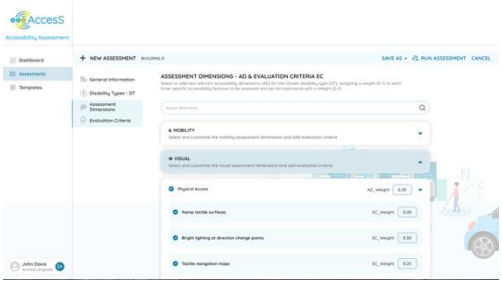
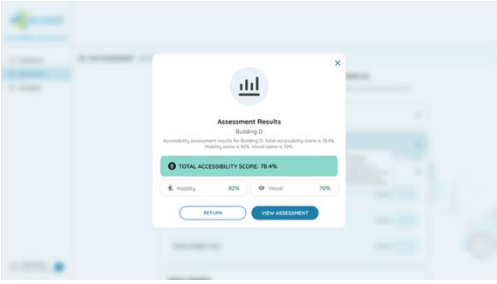
	  • *Frontend, REST API, database, and object storage should operate cohesively as an integrated system.* • *Accessibility scores (total impact score and overall building score) should be computed correctly.* • *Assessment reports have to be securely stored in MinIO and retrievable via the frontend.* • *Authentication and authorization via Keycloak function correctly.* • *System should remain stable and consistent across repeated executions.*

Table 22. AAT Security Test Description

Test Title	*Security Testing of Large-Scale Accessibility Assessment Tool (AAT)*
Identifier	*FORTH-AAT-SEC-01*
Test Type	*Security*
AccesS Suite components	*Large-Scale Accessibility Assessment Tool*
Tester(s)	*FORTH*
Responsible	*FORTH*
Test Purpose	*Assess robustness against unauthorized access, data leakage, and malicious usage.*
Configuration	• ***Web Frontend:*** *Deployed on a web server and accessible externally.* • ***REST API:*** *Deployed as a Docker container and exposed externally through controlled endpoints.* • ***Database:*** *MongoDB used to store the developed Accessibility Assessment Scheme and related configuration data.* • ***Object Storage:*** *MinIO used as an object storage service for assessment reports and generated files.*

<table>
<tr><td></td><td>

- ***Authentication:*** *Keycloak embedded for user authentication and Single Sign-On (SSO) across the Access Suite.*
- ***Deployment & Network:*** *Docker-based deployment with private internal network; only Web Frontend and REST API are exposed externally, while MongoDB, MinIO, and internal services remain isolated.*
- ***Network configuration:*** *Secure communication between containers; HTTPS enabled for external access where applicable.*

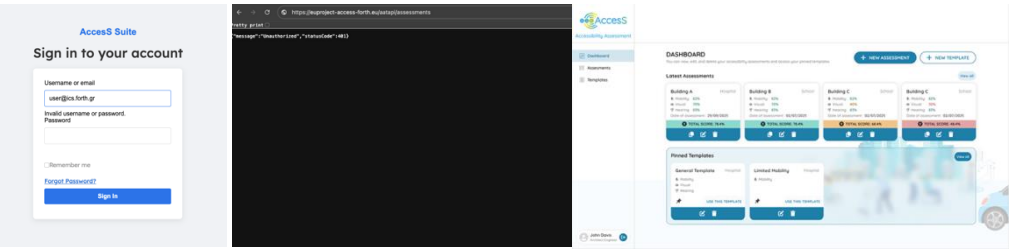
</td></tr>
<tr><td>**Pre-test conditions**</td><td>

- *User roles configured,*
- *Test user accounts available*
- *Secured API endpoints*

</td></tr>
<tr><td>**Test Sequence**</td><td>

<table>
<tr><th>Step</th><th>Type</th><th>Description</th><th>Result</th></tr>
<tr><td>1</td><td>Functional</td><td>Invalid login attempt</td><td>Access denied</td></tr>
<tr><td>2</td><td>Functional</td><td>Unauthorized GUI access</td><td>Access denied</td></tr>
<tr><td>3</td><td>Functional</td><td>Cross-user VUM access</td><td>Access denied</td></tr>
<tr><td>4</td><td>Functional</td><td>Unauthorized API call</td><td>Access denied</td></tr>
<tr><td>5</td><td>Integration</td><td>Malformed input submission</td><td>Safely rejected</td></tr>
</table>

</td></tr>
<tr><td>**Input Data**</td><td>*Authorized/unauthorized requests, malformed JSON payloads*</td></tr>
<tr><td>**Results**</td><td>



Unauthorized access blocked, data confidentiality preserved

</td></tr>
</table>

Enhancing Accessibility and Sustainability in Smart Cities and Smart Buildings: The Universal Accessibility Suite Initiative

www.euproject-access.eu/en/

@EU project AccesS

@EUprojectAccesS

@EUprojectAccesS

This project has received funding from the European Union's Horizon Europe research and innovation programme under the grant agreement number 101147722. The European Union is not liable for any use that may be made of the information contained in this document, which is merely representing the authors' view.