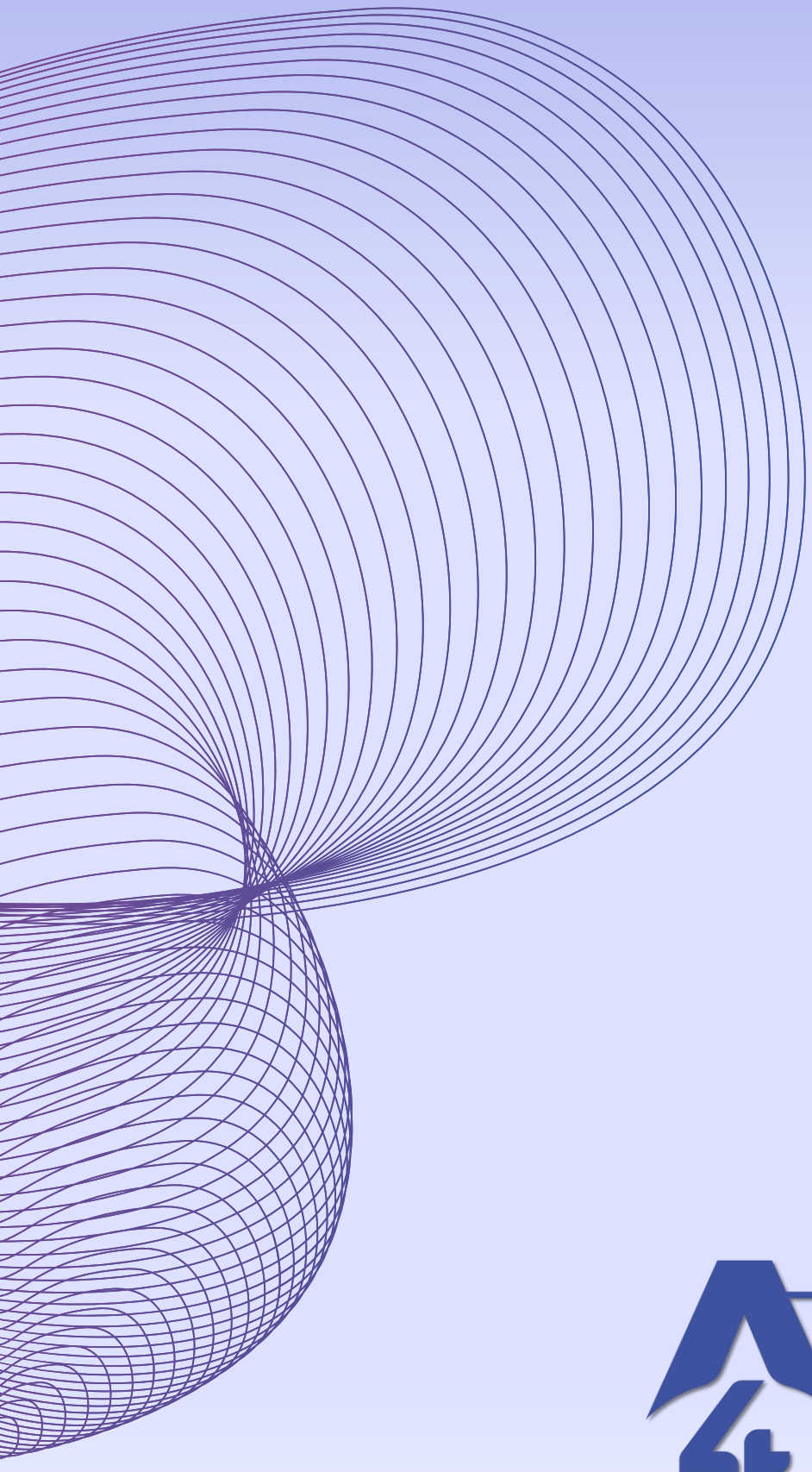


Artificial Intelligence For All

- PROJECT TOOLKIT -

INDEX

Introduction

The AI4ALL project aims to develop a comprehensive self-assessment and management tool specifically designed for educational environments. This innovative platform is tailored to empower both teachers and students by offering an easy-to-use, multilingual, and responsive interface. By addressing the diverse needs of users, the platform facilitates seamless integration into various educational settings. This report provides a detailed overview of the technical choices, development methods, and implementation details involved in creating this robust tool. It covers the entire development process, from initial concept and requirements gathering to final deployment and user training, ensuring a thorough understanding of the platform's capabilities and benefits.

Current AI Landscape in Education

In today's rapidly evolving educational landscape, the integration of Artificial Intelligence (AI) has become increasingly prevalent, revolutionising the way educators approach teaching and students engage with learning (Holmes et al., 2019). This shift towards AI-powered solutions signifies a transformative era in education, where innovative technologies are harnessed to enhance the teaching-learning experience. Let's delve deeper into the current AI landscape in education and explore the multifaceted reasons why educators are embracing AI-powered solutions.

The field of AI in education is experiencing exponential growth, with a diverse array of AI-powered tools and applications catering to various educational needs and objectives (UNESCO, 2021). These tools range from intelligent tutoring systems and virtual classroom assistants to adaptive learning platforms and data analytics tools, showcasing the versatility and potential of AI in education.

1. **Enhance Student Engagement:** AI tools offer interactive and immersive learning experiences that captivate students' attention and foster active participation in the learning process (Holmes et al., 2019). Through gamification, simulations, and virtual reality, AI engages students in dynamic learning experiences that ignite curiosity and promote deeper understanding.
2. **Improve Learning Outcomes:** By personalising learning paths and providing targeted support, AI enhances student comprehension and retention, ultimately leading to improved academic performance. Adaptive learning algorithms analyse student data to identify areas of strength and weakness, delivering customised content and interventions tailored to individual needs.
3. **Automate Routine Tasks:** AI streamlines administrative tasks such as grading, lesson planning, and scheduling, freeing educators to allocate more time and energy to meaningful instructional activities (UNESCO, 2021). Intelligent automation reduces the burden of manual tasks, enabling educators to focus on fostering creativity, critical thinking, and collaboration among students.
4. **Facilitate Data-Driven Decisions:** AI analytics provide educators with valuable insights into student progress, learning preferences, and areas of difficulty,

empowering them to make informed decisions about instructional strategies and interventions. By leveraging data-driven insights, educators can identify trends, track student growth, and optimise teaching practices to maximise learning outcomes.

5. **Support Diverse Learning Styles:** AI-powered adaptive learning platforms cater to individual learning styles and preferences, ensuring that each student receives tailored instruction and support suited to their unique needs and abilities. Personalised learning pathways, interactive tutorials, and real-time feedback mechanisms empower students to learn at their own pace and in their preferred mode of instruction.

In summary, the integration of AI in education offers a multitude of benefits, ranging from increased student engagement and improved learning outcomes to streamline administrative processes and personalised learning experiences tailored to meet the needs of diverse learners. As AI continues to evolve and advance, its role in shaping the future of education is poised to become even more significant, driving innovation, equity, and excellence in teaching and learning worldwide.

Development of the Self-Assessment Tool

In the dynamic landscape of education, the need for comprehensive tools to evaluate and enhance students' digital competencies has become increasingly pronounced. To address this need, the development of a self-assessment tool has emerged as a pivotal endeavour, empowering educators to gauge students' proficiency in key digital skills and tailor instructional strategies accordingly. This section delves into the intricate process of conceptualising, designing, and implementing the self-assessment tool, with a focus on functional and nonfunctional requirements, content development based on the **European Union's Digital Competence Framework (DigComp)**, and the integration of multilingual functionalities. Additionally, insights into the development of the quiz platform, central to the assessment process, will be elucidated, highlighting key features and functionalities aimed at facilitating seamless administration and analysis of assessments. Through meticulous attention to detail and alignment with established frameworks, the self-assessment tool endeavours to serve as a catalyst for promoting digital literacy and empowering students to thrive in an increasingly digital-centric world.

Functional and Non-Functional Requirements of the Platform

In developing the self-assessment tool, meticulous attention was paid to delineating both functional and non-functional requirements, ensuring its effectiveness and usability across diverse educational settings and user demographics.

Functional Requirements

Questionnaire Initialization

The platform starts with a predefined list of questions based on the European Union's Digital Competence Framework (DigComp). This ensures that assessments cover essential digital competencies and align with broader educational goals.

Adaptability

Recognizing the varied nature of professional education, the self-assessment tool was designed to be adaptable, catering to different subject areas and proficiency levels. Whether used in corporate training, vocational programs, or other professional development settings, the tool seamlessly adjusts to meet the specific requirements of each context.

Scalability

With the goal of accommodating a large volume of users and data, the platform was engineered to exhibit scalability without compromising performance. This scalability ensures that the tool can effectively serve educational institutions of varying sizes, from small schools to large universities, while maintaining optimal speed and responsiveness.

Feedback Mechanism

Administrators have continuous access to a dedicated monitoring interface that allows them to track student responses, progress, and quiz status in real-time. This interface provides comprehensive insights into students' performance. Only monitors can view detailed statistics and download quiz data in various formats. For more detailed follow-up, teachers can use the management platform, which offers in-depth analytics and tools for targeted support, reinforcing learning and empowering students to take ownership of their progress.

Non-Functional Requirements

Security

Robust security measures were implemented to safeguard user data and maintain confidentiality. Encryption protocols, access controls, and regular security audits ensure that sensitive information remains protected from unauthorised access or breaches.

Accessibility

Recognizing the diverse needs of users, the platform was designed to be accessible to individuals with varying abilities and needs. Adherence to accessibility standards ensures that all users, including those with disabilities, can effectively utilise the tool without encountering barriers.

Usability

The user interface was thoroughly crafted to be intuitive, with clear navigation and user-friendly features enhancing the overall user experience. Through user testing and feedback iterations, the platform's usability was optimised to accommodate users of all levels of technical proficiency.

Reliability

Ensuring continuous access for users, the platform was engineered to be reliable and available at all times. Redundant systems, failover mechanisms, and proactive monitoring minimise downtime, guaranteeing uninterrupted access to the self-assessment tool for students.

In summary, the self-assessment platform, along with the management platform, provides essential tools for educators and students. By meeting comprehensive requirements, the self-assessment tool serves its purpose well, promoting digital literacy and empowering learners in today's digital world.

Development of the Self-Assessment Tool and Research of a EU Framework

The development of the self-assessment tool was intricately intertwined with extensive research into the European Union's Digital Competence Framework (DigComp), which serves as a comprehensive reference for digital competencies across various domains. This holistic approach ensured that the tool's contents were meticulously curated to align with educational objectives and DigComp's competency areas.

DigComp Framework Integration and Curriculum Mapping

The initial phase involved mapping the contents of the self-assessment tool to the curriculum requirements and learning objectives of the target educational institutions. This mapping ensured that the assessments covered essential digital competencies and aligned with broader educational goals. Building upon the DigComp framework, the contents of the self-assessment tool were structured to encompass the five key competence areas:

- **Information and Data Literacy:** Assessments evaluated students' abilities to articulate information needs, locate and retrieve digital data, and judge the relevance of sources. Skills in managing and organising digital data and content were also assessed.
- **Communication and Collaboration:** Proficiency in interacting, communicating, and collaborating through digital technologies was evaluated. This included managing digital presence, identity, and reputation, as well as participating in digital societies.
- **Digital Content Creation:** Students' skills in creating and editing digital content, integrating information into existing knowledge bases, and understanding copyright and licensing principles were assessed.
- **Safety:** Assessments addressed students' knowledge and practices related to protecting devices, content, personal data, and privacy in digital environments. Awareness of digital technologies for social well-being, inclusion, and environmental impact was also evaluated.
- **Problem Solving:** Questions assessed students' abilities to identify needs and problems, conceptualise solutions, and innovate using digital tools. Staying updated with digital advancements and adapting to evolving challenges were key components.

Curriculum Mapping and Threshold Definition

Collaborative sessions were conducted with educators to outline in detail grading thresholds and formulate final score formulas. These interactive discussions encompassed the

delineation of proficiency criteria within each competency area, alongside the establishment of performance benchmarks. Leveraging DigComp proficiency levels, thresholds were rigorously calibrated to ensure both consistency and alignment with established frameworks. The development of final score formulas was methodically crafted to amalgamate scores across competency domains, thereby furnishing a comprehensive assessment of students' digital competence.

Content Validation and Continuous Monitoring

The content framework underwent thorough validation by experts and educators familiar with DigComp, refining its relevance, accuracy, and alignment with educational objectives. Collaboration with experts in digital literacy and education further validated the tool's alignment with DigComp, enhancing its credibility and utility. Continuous monitoring ensured the tool remained aligned and relevant amidst evolving educational landscapes.

Questions Development

The development of assessment questions for the self-assessment tool involved a meticulous and systematic process aimed at thoroughly evaluating students' proficiency in key digital competencies outlined in the DigComp framework. This multifaceted process included several essential steps:

Identification of Competency Areas and Granularity Levels

This pivotal phase involved thoroughly analysing the DigComp framework to pinpoint the overarching competency areas, each encompassing a spectrum of granular proficiency levels. For instance, within Domain 1: Information and Data, the sub-competencies were delineated into tasks such as Navigating, Searching, and Filtering Data, Evaluating Data and Information, and Managing Digital Data.

Once these competency areas were identified, further depth was achieved by dissecting each sub-competency into specific skill sets and knowledge domains. For example, within Domain 3: Digital Content Creation, the sub-competency of "Creating Digital Content" may involve skills such as graphic design, multimedia production, and content formatting, each representing a distinct level of proficiency.

This thorough process of identifying competency areas and granularity levels ensured that the assessment questions could effectively evaluate students' proficiency across a wide spectrum of digital skills and knowledge domains. It provided a structured framework for the development of assessment questions tailored to address specific competencies and their corresponding proficiency levels within each domain.

Question Formulation

Each assessment question was carefully crafted to holistically assess students' knowledge, skills, and attitudes across the spectrum of digital competencies. Special attention was given to ensuring that the questions were not only clear and concise but also aligned with the proficiency levels stipulated in the DigComp framework, thus guaranteeing the relevance of the assessment.

Diversity and Relevance

To ensure the comprehensiveness and relevance of the assessment, a diverse range of topics and scenarios relevant to students' digital experiences were incorporated into the questions. Real-life situations, case studies, and practical examples were woven into the fabric of the assessment, enriching its authenticity and fostering deeper engagement among students.

Validation and Review

Following the formulation of the initial set of assessment questions, they underwent a rigorous validation and review process by subject matter experts and educators who possessed a deep understanding of the DigComp framework. Feedback from these stakeholders was carefully analysed and incorporated to refine and enhance the questions further. This iterative process ensured that the questions maintained their accuracy, clarity, and efficacy in assessing digital competencies effectively.

Enhancing User Experience through Multilingual Capabilities

Recognizing the significance of multilingual accessibility, our team prioritised enhancing the self-assessment tool to accommodate users from diverse linguistic backgrounds. Here's how we achieved this:

Translation and Contextualization

We undertook a meticulous process of translating assessment questions and interface elements into multiple languages, ensuring inclusivity for users who may not be proficient in the primary language. Additionally, we focused on contextualization to ensure that the translations were not only accurate but also natural-sounding and relevant in the target language, enhancing user comprehension and engagement.

Verification and Adaptation

To uphold the integrity of our translations, project partners thoroughly verified each translation to guarantee relevance. Furthermore, translations were thoughtfully adapted to seamlessly integrate into the context of quizzes and assessments, maintaining consistency and coherence across languages.

File Preparation

Our team diligently prepared files containing translated text and interface elements in formats compatible with the self-assessment tool's architecture and design. This precise preparation ensured smooth integration and functionality, regardless of the language selected by the user. Different file formats were utilised and tailored to meet the specific requirements of the platform.

Integration and Implementation

Translated files were seamlessly integrated into the backend of the self-assessment tool, enabling dynamic language switching based on user preferences. Through efficient coding practices, we facilitated the retrieval and display of translated text and interface elements during runtime, providing users with a harmonious multilingual experience.

User Training and Support

In our commitment to facilitating optimal utilisation of multilingual functionalities, we developed comprehensive user training and support materials for educators and students. These resources, including tutorials, guides, and help documentation, offered clear instructions for accessing and navigating the tool in different languages, empowering users to make the most of its capabilities.

Development of the Self-Assessment Management Tool

The development of the self-assessment management tool represents a transformative leap forward within the AI4ALL initiative. Positioned as a pivotal cornerstone in the project's mission, this tool offers educators a multifaceted platform that is designed to not only monitor but also analyse and optimise the intricate tapestry of student learning experiences.

Harnessing the prowess of cutting-edge web development technologies and the ingenuity of advanced data analysis techniques, this tool stands as a beacon of innovation within the educational landscape. Through the seamless integration of dynamic web technologies like Python, Flask, PostgreSQL, and JavaScript, educators are granted a level of accessibility and functionality that is both robust and intuitive.

At its core, this robust framework serves as a catalyst for transformation, facilitating the ideal fusion of machine learning models and data analysis libraries. This convergence empowers educators with a suite of powerful tools, equipping them to navigate the complexities of teaching and learning with precision and efficacy.

Educators find themselves met with a dynamic and user-friendly interface upon starting their use of the self-assessment management tool. Through every interaction, they gain access to valuable insights and resources designed to enhance each student's educational experience.

In summary, the self-assessment management tool goes beyond being just a piece of technology. It reflects a commitment to innovation and empowerment, showcasing the dedication of the AI4ALL project to improving education for all and fostering inclusivity.

Functional and Non-Functional Requirements of the Platform

Functional Requirements

Web Development and Integration

The self-assessment management tool is developed using a robust combination of Python, Flask, PostgreSQL, and JavaScript. Python and Flask form the backend framework, providing a scalable infrastructure capable of handling a large volume of users and data. PostgreSQL serves as the database, ensuring reliable and efficient data storage and retrieval. JavaScript powers the frontend, creating dynamic and interactive user interfaces. Additionally, the platform integrates machine learning models and data analysis libraries, such as R and various Python-based libraries, to offer advanced educational insights.

The web application is hosted on internal servers, ensuring that data remains secure and under control. Regular backups and maintenance activities are performed to keep the system

current and reliable. Rebuilding the system involves setting up the environment, including installing Python, Flask, and PostgreSQL on the server, and configuring a virtual environment for dependency management. The next steps involve cloning the project repository from the internal version control system and installing the required Python packages using pip. After installing and configuring PostgreSQL, a web server like Nginx or Apache is set up to serve the Flask application and route requests appropriately. Finally, the application is deployed, and the Flask app is run, ensuring it is accessible via the configured web server.

User Authentication and Management

The platform includes secure log-in and log-out mechanisms to protect user accounts and sensitive data. User roles are differentiated into three main categories: administrators, teachers, and students. Administrators have the ability to monitor registered users and ensure system integrity. Teachers can create and manage quizzes, view detailed analytics, and monitor student progress. Students can take quizzes and view their own progress and feedback. This role-based access ensures that each user type has the appropriate level of access and functionality within the platform.

Real-time Monitoring and Feedback

Teachers have continuous access to a monitoring interface where they can track student responses, progress, and quiz status in real-time. This interface offers comprehensive insights into students' performance and responses. The detailed analytics available on the management platform provide an in-depth analysis of student performance and progress, enabling teachers to offer targeted support and interventions. Administrators can oversee the overall system and ensure that user activities are in compliance with platform policies.

Data Analysis and Reporting

The platform boasts sophisticated analytics and reporting functionalities, equipping educators with invaluable insights into student performance. Through careful aggregation and analysis of data, trends and areas for improvement are systematically identified, empowering educators to tailor their teaching strategies accordingly. One notable feature is the platform's support for exporting data exclusively in the widely-used XLSX format, ensuring compatibility and ease of integration with other systems.

By offering this flexibility, educators can seamlessly incorporate the platform's insights into their existing workflows, maximising the utility of the data collected. Whether it's for in-depth analysis or absolute integration with other educational tools, the ability to export data in XLSX format streamlines the process, making it more accessible and actionable for educators.

Non-Functional Requirements

The non-functional requirements of the self-assessment management tool are critical in ensuring that the platform not only meets the technical specifications but also provides a secure, scalable, accessible, user-friendly, and reliable experience for all users.

Security

Security is paramount in the design and operation of the self-assessment management tool. The platform employs robust encryption protocols to protect sensitive data, both at rest and in transit. Secure access controls are implemented to ensure that only authorised users can

access specific functionalities and data. Multi-tiered authentication processes and role-based access control (RBAC) further enhance security by restricting access based on user roles. Additionally, session timeouts are enforced to minimise the risk of unauthorised access from unattended sessions. By prioritising security, the platform safeguards the integrity and confidentiality of user data, fostering trust among educators and students.

Scalability

The platform is designed to handle increasing user load and data volume without compromising performance. Scalability is achieved through a modular architecture that allows for the seamless addition of resources as demand grows. Techniques are implemented to optimise performance and prevent any single server from becoming a bottleneck. This capability is crucial for accommodating varying usage patterns, such as spikes in user activity during peak times. By ensuring that the platform remains responsive and efficient under varying loads, scalability supports a growing user base and an expanding dataset, making the tool robust and future-proof.

Accessibility

The platform aims to adhere to international accessibility standards, such as the Web Content Accessibility Guidelines (WCAG), to better accommodate users with varying abilities. This commitment involves designing the user interface and experience to be more inclusive of individuals with disabilities. While not all accessibility features are fully implemented, efforts are made to test the platform with diverse user groups to identify and address accessibility barriers. By prioritising accessibility, the platform strives to ensure that most users, regardless of their abilities, can interact with and benefit from the self-assessment tool.

Usability

An intuitive user interface with clear navigation is essential for ensuring that the platform is user-friendly. The design process involves extensive user research and testing to understand the needs and preferences of educators, students, and administrators. Simplified workflows and logical information architecture make it easy for users to navigate the platform and complete tasks efficiently. User feedback is continually solicited and incorporated to refine the interface and improve the overall user experience. By focusing on usability, the platform enhances user satisfaction and engagement, making it a valuable tool for teaching and learning.

Reliability

High uptime and reliability are achieved through the implementation of redundant systems and proactive monitoring. Redundancy involves duplicating critical components and systems so that a backup is available in case of failure. This approach minimises downtime and ensures continuous access to the platform. Proactive monitoring involves real-time tracking of system performance and health metrics, enabling prompt detection and resolution of issues. Automated alerts and routine maintenance schedules further contribute to maintaining high reliability. By ensuring that the platform is dependable and consistently available, reliability supports uninterrupted access to the self-assessment tool, enhancing its effectiveness and trustworthiness.

In summary, the non-functional requirements of security, scalability, accessibility, usability, and reliability are integral to the success of the self-assessment management tool. By adhering to these requirements, the platform not only meets its technical objectives but also provides a secure, inclusive, and high-performing environment that enhances digital literacy and supports personalised education.

Basic Structure of the Platform

Crafted with the user in mind, the platform's fundamental architecture aims to deliver an intuitive and accessible experience to both educators and students. This structured design encompasses various essential elements, each meticulously tailored to enable effortless navigation and interaction throughout the platform's ecosystem.

Pages/Sections

The platform encompasses distinct pages and sections, each strategically designed to cater to different user roles and functionalities. These include:

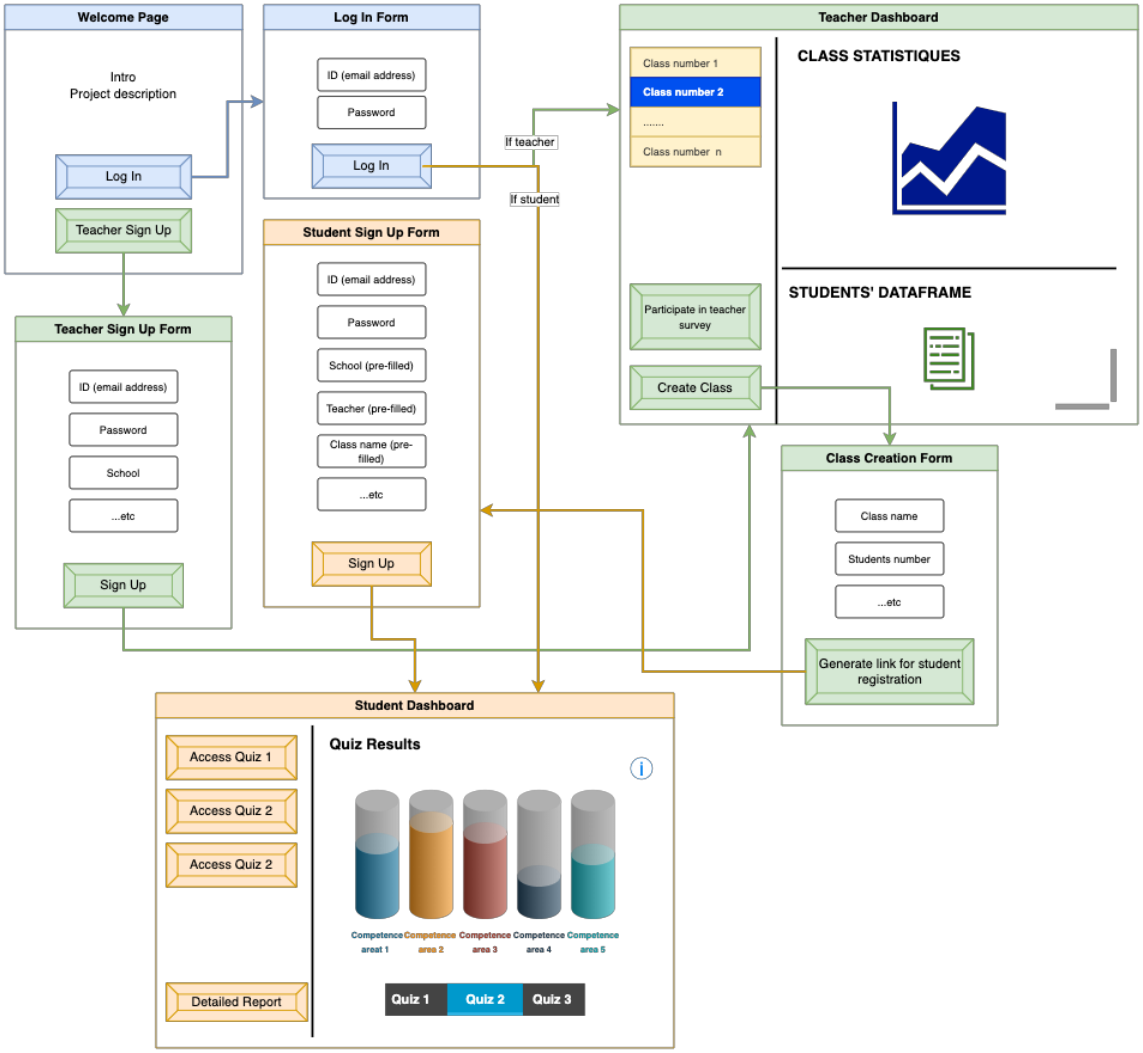
1. **Teacher Dashboard:**
    - Class Creation/Management: Enables teachers to create and manage classes, assign assessments, and monitor student progress.
    - Global and Class Statistics: Provides comprehensive statistics and analytics on student performance at both class and global levels, facilitating informed decision-making.
    - Student Table: This section serves as a centralised hub where educators can access detailed information about each student enrolled in their classes. It provides a comprehensive summary of student data, including their names, progress, and performance metrics across various self-assessment modules. The table offers a convenient and organised overview of student performance, allowing educators to quickly identify trends, strengths, and areas needing improvement. By visualising student progress and performance in a structured format, educators can efficiently track individual achievements and tailor their instructional strategies to meet the diverse needs of their students.
    - Machine Learning Insights and Recommendations: Offers valuable metrics and recommendations derived from machine learning models. It provides educators with a comprehensive overview of the current performance of the machine learning algorithms employed within the platform. These insights include metrics such as accuracy, allowing educators to assess the effectiveness of the models in analysing student data. Additionally, this section offers tailored recommendations based on the analysis of student performance, helping educators refine their teaching strategies and interventions to better meet the needs of individual students.
2. **Student Dashboard:**
    - Self-Assessment Access: The Student Dashboard provides students with convenient access to self-assessment quizzes specifically tailored to their learning objectives and competency levels. This feature enables students to engage in personalised learning experiences that target their individual needs and areas for improvement.

- Track Progress: The Student Dashboard allows students to track their progress over time by providing visual representations of their performance across various assessments. This feature empowers students to monitor their academic growth and take proactive steps to enhance their learning outcomes.

3. **Admin Dashboard:**
    - Teacher and Student Information: The dashboard provides detailed tables that display essential information about teachers and students, including names, contact details, and roles. This centralised view enables administrators to quickly access pertinent information and efficiently manage user accounts.
    - Total Number of Registered Individuals: Administrators can easily track the total number of registered users, including both teachers and students. This metric provides valuable insights into the platform's user base and growth trajectory, facilitating strategic decision-making and resource allocation.
    - Country Filters: To enhance data visibility and analysis, the dashboard features intuitive filters that allow administrators to segment user data by country. This functionality enables administrators to gain insights into user demographics and geographical distribution, facilitating targeted interventions and support initiatives.

Illustration: Platform Architecture Diagram

This diagram shows the overall architecture of the platform, highlighting the main components such as the Teacher Dashboard, Student Dashboard. It also illustrates the flow of data between these components and how they interact with each other.



Log-in/Out, Language Choice, Accessibility and Responsiveness

Log-in/Out Functionality

The log-in/out functionality provides users with a secure pathway to access the platform using their unique credentials. Once logged in, users are greeted with a personalised dashboard tailored to their specific role and preferences. While multi-factor authentication (MFA) is not currently integrated, stringent security measures are in place to safeguard user accounts and data. Additionally, the log-in feature includes useful functionalities such as password recovery, ensuring that users can easily regain access to their accounts if needed.

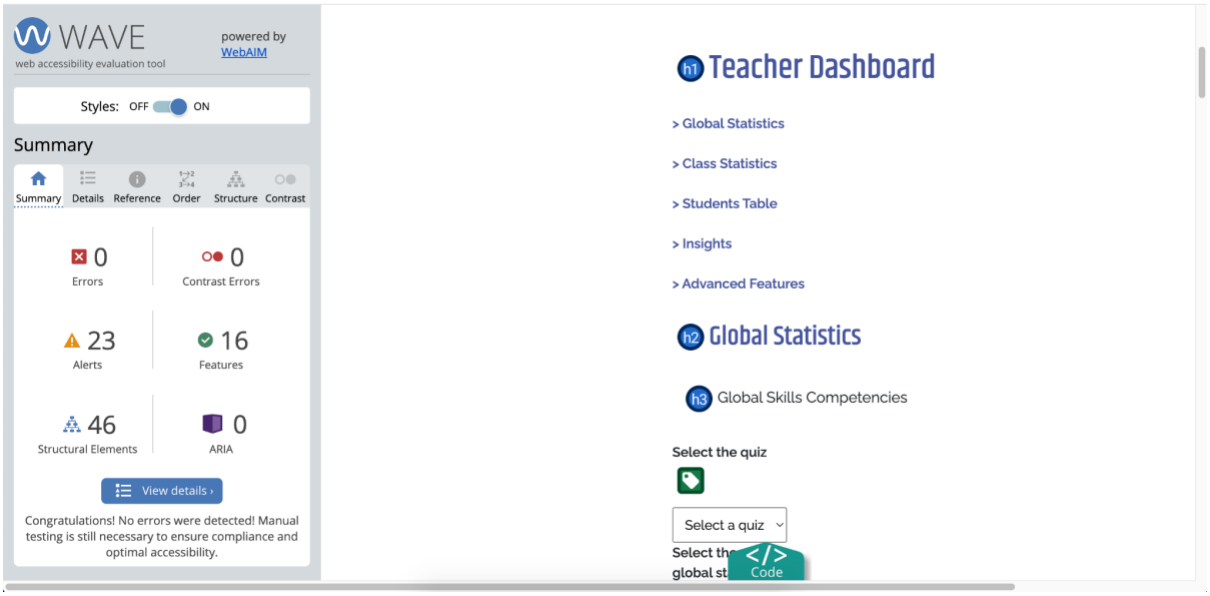
Language Choice

Language choice is conveniently accessible within the navigation menu, offering users the flexibility to select their preferred language. Whether during the log-in process or within their account settings, users can effortlessly switch to their native language, enhancing their overall experience on the platform. This emphasis on language accessibility and user-friendly features underscores the platform's commitment to promoting inclusivity and ease of use for all users.

Developing Accessibility with WAVE Tool

The platform underwent rigorous evaluation using accessibility tools such as the WAVE tool to ensure adherence to international standards like the Web Content Accessibility Guidelines (WCAG). Here's how identified errors were systematically addressed:

1. Identifying Accessibility Issues: The WAVE tool scanned the platform and highlighted various accessibility barriers, including missing alternative text for images, improper use of heading tags, colour contrast issues, and interactive elements lacking proper labels.
2. Prioritization and Planning: Based on the WAVE tool's findings, accessibility issues were prioritised according to their impact and importance. This step involved categorising issues into critical, high-priority, and medium-priority fixes.
3. Implementing Fixes: Development teams collaborated to implement fixes for identified issues:
    - Alt Text for Images: All images across the platform were reviewed and provided with descriptive alternative text to ensure they were accessible to users relying on screen readers.
    - Heading Structure: Proper heading tags (h1, h2, etc.) were applied consistently throughout the platform's content hierarchy to aid navigation and improve readability.
    - Colour Contrast: Elements with insufficient colour contrast were adjusted to meet WCAG standards, ensuring text and interactive elements were distinguishable for users with visual impairments.
    - Form Labels and ARIA Roles: Interactive elements like form fields and buttons were enhanced with proper labels and ARIA (Accessible Rich Internet Applications) roles to clarify their purpose for assistive technologies.
4. Testing and Validation: Post-implementation, the platform underwent thorough testing with assistive technologies and by real users with disabilities. Feedback from these tests guided further refinements and ensured the effectiveness of accessibility improvements.
5. Continuous Improvement: Accessibility remains an ongoing commitment. Regular audits, user feedback, and updates to accessibility guidelines drive continuous improvement efforts to make the platform more inclusive for all users.



By leveraging tools like the WAVE tool and following a structured approach to accessibility, the platform aims to provide an inclusive user experience, enabling users of diverse abilities to fully engage with its features and content.

Responsive Functionalities

The platform is designed to be responsive across various devices and screen sizes, ensuring an optimal user experience regardless of the device being used. Whether accessed from a desktop computer, laptop, tablet, or smartphone, users can expect consistent functionality and seamless navigation. Responsive design principles are implemented to adapt the layout and content dynamically, providing users with an intuitive and visually appealing interface. This responsiveness enhances accessibility and usability, allowing users to engage with the platform conveniently from anywhere, at any time.

Integrating User Feedback into Platform Design

The platform's design has been thoroughly tailored by developers in response to user feedback and specific requirements. This customised design ensures that the platform aligns with users' needs and preferences without requiring direct CSS customization. By incorporating user feedback into the design process, developers have crafted an interface that enhances user experience and engagement.

The design adjustments include modifications to colours, fonts, layouts, and styles, creating a visually appealing and user-friendly interface. These changes not only meet users' expectations but also reflect the branding guidelines and usability standards of the organisation. By continuously refining the platform's design based on user feedback, developers ensure that it remains relevant, intuitive, and effective for all users.

Through collaborative efforts between developers and users, the platform's design evolves to meet evolving needs and expectations. This approach emphasises user-centric design principles, resulting in a platform that prioritises user satisfaction and productivity.

CSS Styles for Desktop

For a closer look at CSS Styles for Desktop, please refer to the annexes for code snippets.

Variables/Features Definition

Variables or features in the context of this study refer to the specific characteristics or attributes of the data under analysis. These features play a crucial role in modelling and understanding the underlying patterns or relationships within the data. In this section, we provide a comprehensive definition and description of each variable or feature utilised in our analysis. Understanding these variables is essential for interpreting the results and drawing meaningful conclusions from our study.

Self-Assessment Results

The self-assessment results encompass feedback and evaluations provided by participants regarding their knowledge and proficiency across various competence areas outlined in the Digital Competence Framework (DigComp). Participants were asked to assess their competency levels in key digital skills such as information and data literacy, communication and collaboration, digital content creation, safety, and problem-solving. By soliciting self-reported assessments, this section aims to capture participants' perceptions of their digital competencies and proficiency levels. Analysing these self-assessment responses provides valuable insights into participants' perceived strengths and areas for improvement in relation to digital skills and competencies as defined by DigComp.

Demographic Information

Demographic information encompasses a range of socio-economic and socio-cultural factors that characterise the study participants' backgrounds and contexts. These include variables such as age, gender, education level, occupation, and geographical location, among others. Analysing demographic information provides valuable context for understanding how different population segments engage with the subject matter of the study and enables researchers to identify potential patterns or disparities that may exist within the data. In this section, we provide a comprehensive overview of the demographic profile of the study participants, shedding light on the diversity and representation within the sample population.

Registration Process Implementation

The registration process implementation delineates the systematic steps and procedures employed to onboard users onto the digital platform. This encompassing process typically begins with the creation of user accounts, where prospective users provide essential information such as personal details, contact information, and often create unique login credentials. Additionally, this phase may involve the validation of user-provided data to ensure

its accuracy and authenticity. The registration process implementation is meticulously designed to streamline user onboarding while maintaining data integrity and security.

Email Verification Process

The email verification process constitutes a critical component of user authentication and security measures within the digital platform. Upon registering for an account, users are usually required to verify their email addresses to confirm their identities and establish the legitimacy of their accounts. This verification often entails the transmission of a verification link or code to the user's provided email address, which they must then click or enter to complete the verification process. By validating email addresses, the platform enhances security measures, mitigates the risk of fraudulent accounts, and ensures the integrity of user data.

Teacher and Student Registration

The registration process accommodates both teachers and students, each with distinct functionalities and privileges tailored to their roles within the digital learning environment. Teacher registration typically entails additional steps beyond basic user registration, including educational qualifications, and years of experience. Once verified, teachers gain access to specialised tools and features for course creation, management, and assessment. Conversely, student registration focuses on capturing essential information such as professional background details, academic affiliation, and educational objectives.

By facilitating separate registration pathways for teachers and students, the platform optimises user experience and ensures that each user group receives tailored support and resources aligned with their respective roles and responsibilities.

Activation of Quizzes

Activating quizzes from the teacher's dashboard is a straightforward process. This feature allows teachers to make self-assessments visible to students at the most opportune times, aligning with their course programmes. By doing so, teachers can ensure that students are adequately prepared and have covered the necessary material before attempting the quiz. The flexibility to activate quizzes also enables teachers to incorporate a variety of assessment strategies, such as formative or summative evaluations, into their teaching methodology. This way, students can gauge their understanding, identify areas for improvement, and receive timely feedback, all of which are integral to a successful learning experience.

Data Protection

In accordance with the General Data Protection Regulation (GDPR), we set a policy to treat the data as follows:

Teacher data: As soon as they establish their profile, they receive an email confirming it, at which point they permit the AI4All administrator to keep their data solely for statistical purposes.
Student or SME workers' Data: When they receive a link from their teacher or instructor to create their profile and begin using the platform, they must check a box allowing their teacher

to access their data and treat it with strict confidentiality in order to exclusively address educational activities.

Statistics Development

Statistics development on the platform involves the creation and implementation of algorithms and models that process educational data to generate meaningful insights. This includes the collection and analysis of data related to student performance, engagement, and progress. Advanced statistical methods and machine learning techniques are employed to identify patterns, trends, and correlations within the data. The development process ensures that the statistics generated are accurate, reliable, and actionable, providing educators with valuable tools for decision-making. By continuously refining these statistical models, the platform can offer increasingly sophisticated analytics that support personalised learning, early intervention, and overall educational improvement.

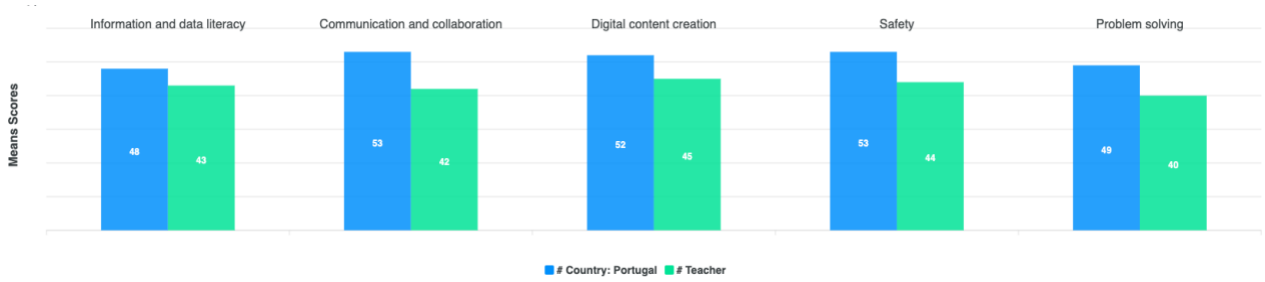
Global and Class Statistics

Global and class statistics provide educators with a comprehensive overview of student performance at different levels.

Global Statistics

These offer a macro view, aggregating data across all classes and students on the platform. This includes metrics about average scores. Global statistics help educators identify broader trends, benchmark performance against larger datasets, and make informed decisions about curriculum and policy adjustments.

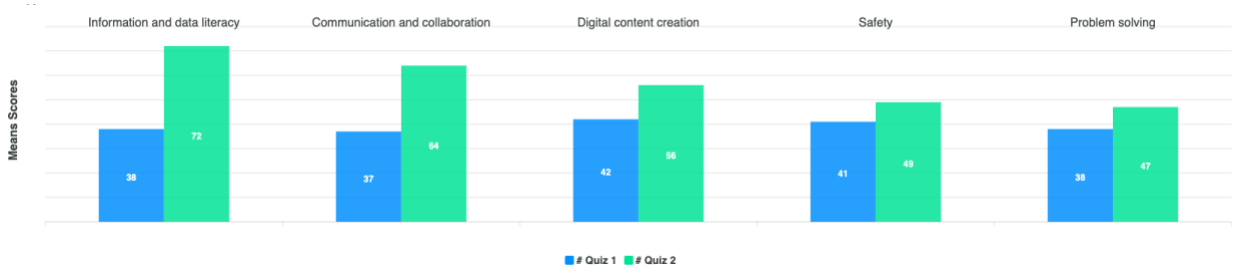
Here's an illustration comparing the average scores per competence area of one teacher's students against the average scores of all students in Portugal. This comparison provides insights into performance differences across various competence areas within a specific country context.



Class Statistics

In contrast, class statistics focus on specific classes or groups of students, providing detailed insights into the performance and engagement of a particular cohort. This includes data about individual student scores. Class statistics enable teachers to identify classes that may need additional support, track the effectiveness of their teaching strategies, and tailor instruction to meet the needs of their students. By leveraging both global and class statistics, educators can achieve a balanced perspective, optimising both overall educational outcomes and individual student success.

Here's an illustration comparing the performance of students from the same class in Quiz 1 against Quiz 2. This comparison offers insights into the progression of student learning and any changes in performance between the two quizzes.



Download of Class Results Data

Teachers have the capability to download detailed class results data in the form of an Excel spreadsheet (xlsx file). This feature allows educators to access comprehensive information about their students and their performance. The downloaded file includes a wide array of data, such as student registration details (e.g., names, IDs, and contact information), as well as detailed quiz results.

The quiz results section provides an in-depth look at each student's performance on individual quiz items, including scores for each question, scores per competency and competence area as well as overall quiz scores. This granular level of detail enables teachers to analyse student performance at a micro level, identifying specific areas where students may be struggling or excelling.

By utilising this downloadable data, teachers can easily conduct offline analysis, prepare reports, and tailor their instructional strategies to better meet the needs of their students. The ability to download and review this data enhances the overall teaching and learning experience, providing valuable insights that support data-driven decision-making and personalised education.

Models of AI

The educational platform employs various AI models to enhance the learning experience and provide actionable insights. These models leverage advanced machine learning (ML) techniques to analyse data, predict outcomes, and generate recommendations. Among the diverse types of models used, there are standard models tailored for general application across the platform and customizable models that allow for user-specific configurations. Both types serve distinct purposes in driving the efficacy of the platform’s educational tools.

Standard Model for Teacher Dashboard

The standard model integrated into the teacher dashboard is designed to classify students into predicted performance levels for an upcoming quiz (Quiz 2) based on information from their registration details and their performance in the previous quiz (Quiz 1). This classification model uses a sophisticated algorithm that analyses various input features such as

demographic information, prior academic performance, and specific scores from Quiz 1 to predict how students will perform in Quiz 2.
The model categorises students into five distinct levels based on their predicted scores:

- Basic (25-44%): At this level, students will have a foundational understanding of digital concepts. They will possess minimal digital skills necessary for basic tasks, indicating they might need significant support and foundational instruction to progress further.
- Intermediate (45-64%): Students at this level demonstrate a moderate understanding and ability to apply digital skills in various contexts. They can perform tasks independently but might struggle with more complex problems, indicating a need for more targeted practice and reinforcement of concepts.
- Proficient (65-79%): Students classified as proficient have a solid grasp of digital concepts and can effectively apply their skills in a wide range of situations. They show problem-solving abilities and can work both independently and collaboratively, indicating readiness for more challenging tasks and advanced applications.
- Advanced (80-89%): At the advanced level, students exhibit a high level of digital competence. They can creatively solve complex problems, utilise digital tools effectively, and contribute to digital project development, demonstrating significant mastery and the ability to handle sophisticated tasks.
- Expert (90-100%): The highest level, expert, is reserved for students with exceptional digital competence. These students can lead digital projects, innovate, and apply their skills to professional-level tasks. They are also capable of teaching others and contributing significantly to digital literacy in broader communities, reflecting their exceptional proficiency and leadership in digital environments.

This predictive classification allows teachers to identify students' potential performance levels in advance, enabling them to tailor their instructional strategies and provide targeted support where needed.

The following script encompasses a comprehensive pipeline for preprocessing, training, evaluating, and saving machine learning models tailored for educational data analysis. Leveraging libraries such as pandas, NumPy, and scikit-learn, it provides an end-to-end solution for data processing, model training, and interpretation.

Report on Model Implementation

On a technical level, our Python script orchestrates the preprocessing and training of machine learning models using a dataset containing student information. It employs a suite of widely-used libraries such as pandas, numpy, scikit-learn, and shap. Below, we outline the core steps involved in handling the standard machine learning model on the backend:

1. Importing Libraries: The script initiated by importing necessary libraries. This sets the stage for leveraging advanced boosting techniques in model training.
2. Custom Transformer Classes: Two custom transformer classes, `MeanQuizTransformer` and `AssignGroupTransformer`, are defined. These classes are instrumental in computing the mean of quiz scores and assigning students to groups based on their mean scores, respectively. Custom transformers like these facilitate tailored data transformations as per the specific requirements of the dataset.
3. Preprocessing Data: The `preprocess_data` function takes charge of preprocessing the input data. It performs a series of crucial tasks including checking the data format (numpy array or pandas DataFrame), handling missing values, and applying

preprocessing steps for both numeric and categorical features. This is achieved through scikit-learn's powerful ColumnTransformer and Pipeline utilities, ensuring consistency and efficiency in data processing.

4. Generating Features: The `preprocess_data_and_generate_features` function filters columns, calculates the mean of quiz scores, and assigns students to groups using the custom transformer classes. Feature generation is a pivotal aspect of model preparation, allowing for the creation of meaningful representations of the input data that capture relevant patterns and insights.
5. Splitting Data for Training: The `split_data_and_prepare_for_training` function divides the preprocessed data into training and testing sets using scikit-learn's StratifiedShuffleSplit. This ensures that the distribution of target classes remains balanced across both sets, crucial for robust model evaluation.
6. Training Models: The `train_models` function undertakes the training of four distinct machine learning models—GradientBoostingClassifier,, RandomForestClassifier, and DecisionTreeClassifier—using scikit-learn's Pipeline and ColumnTransformer. By encapsulating preprocessing and modelling steps within a unified pipeline, this approach streamlines the training process and enhances reproducibility.
7. Evaluating Models: The `evaluate_models` function assesses the performance of the trained models on the testing data, computing accuracy metrics and confusion matrices using scikit-learn's `accuracy_score` and `confusion_matrix`. This crucial step provides insights into how well the models generalise to unseen data and their ability to correctly classify students into predefined groups.
8. Extracting Feature Importance: The `extract_feature_importance` function extracts feature importances from the trained models, where available. Understanding feature importance aids in interpreting model predictions and identifying the most influential factors driving student group assignments.
9. Saving and Loading Models: The script provides functionality for saving and loading trained models, along with utility functions for calculating accuracy and confusion matrix. This enables seamless integration of model deployment and inference pipelines, facilitating model reuse and scalability.
10. Calculating SHAP Values: Finally, the `get_shap_values` function computes SHAP (SHapley Additive exPlanations) values for the trained models using the shap library. SHAP values offer insights into the individual contributions of features towards model predictions, fostering interpretability and trust in the model outcomes.

By encapsulating these steps in a structured and modular manner, the script empowers efficient preprocessing of data, model training, and generation of actionable insights from the student dataset. This approach ensures that both the back-end system and teacher dashboard benefit from accurate and reliable machine learning models. With clear delineation of tasks and seamless integration of SHAP values for interpretability, the script enhances the overall performance and usability of the system, enabling informed decision-making and effective educational interventions.

Analysing Model Performance with ROC AUC Curve

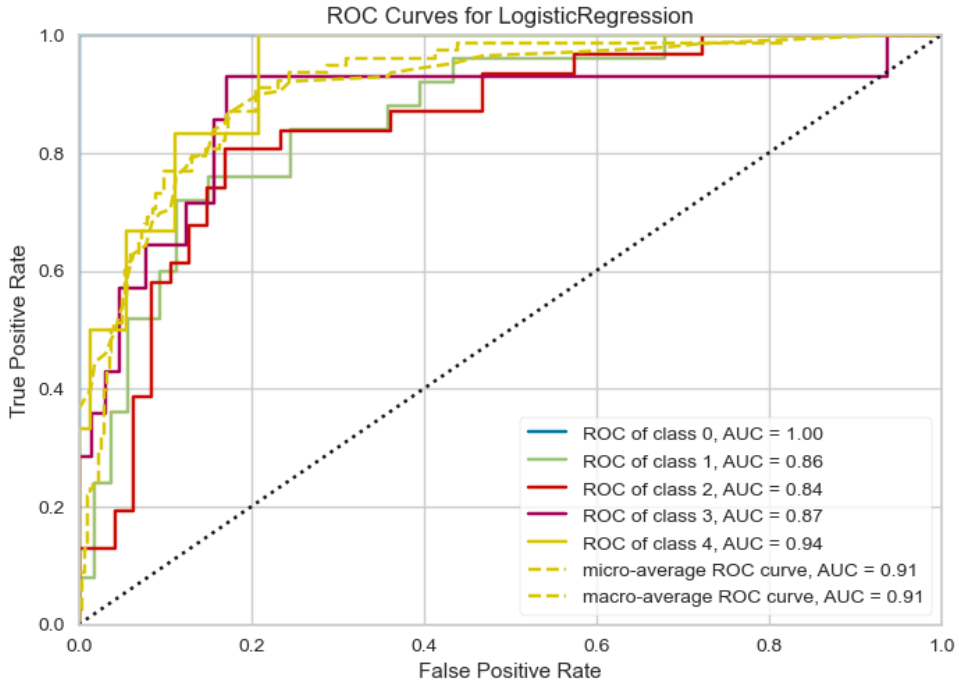
Upon completing the aforementioned steps, we turn our attention to evaluating the performance of the trained models using the Receiver Operating Characteristic (ROC) curve

and the Area Under the Curve (AUC) metric. This analytical approach provides valuable insights into the models' ability to distinguish between positive and negative classes.

The ROC curve showcases the relationship between sensitivity (true positive rate) and specificity (true negative rate) across different threshold settings. Ideally, a model with perfect discrimination would yield an ROC curve that ascends to the upper-left corner, indicating high sensitivity and low false positive rate at all thresholds.
The AUC value serves as a summary statistic for the ROC curve, representing the overall performance of the model. A higher AUC signifies better discrimination between positive and negative classes.

In the graph below, we present the ROC curve for our standard trained model, including logistic regression. Each curve is labelled with its corresponding AUC value, providing a quantitative assessment of the model's discriminatory capability. By visually comparing these curves, we can discern the effectiveness of each classifier and determine the optimal choice for our task. Notably, the minimum AUC value among the models exceeds 0.8, indicative of strong discriminatory performance.

The numeric classes ranging from 0 to 4 represent different predicted groups corresponding to students' performance levels on Quiz 2, spanning from Basic to Expert proficiency levels. This comprehensive analysis enables us to make informed decisions regarding model selection and performance evaluation.



Custom Model Training

The custom model training feature allows educators to build machine learning models tailored to their specific needs and educational contexts. This section provides an overview of how the custom model works, from data preparation to model evaluation.

How the Custom Model Works

Data Preparation

The custom model training process begins with data preparation, where the selected features and target variables are compiled into a training dataset. This dataset includes all relevant student data points that will be used to train the model. Effective data preparation ensures that the model has access to high-quality, relevant information, which is essential for generating accurate predictions.

Feature Engineering

Feature engineering plays a critical role in this phase. Users can apply various techniques to enhance the dataset, such as:

- Normalisation: Standardising data to reduce variability and improve model performance.
- Transformation: Converting features to more suitable formats, like applying logarithms to skewed data.

Additionally, a crucial aspect of feature engineering is implementing a train-test strategy. This involves splitting the dataset into two subsets:

- Training Set: Typically 70-80% of the data, used to train the model and learn the underlying patterns.
- Testing Set: The remaining 20-30%, used to evaluate the model's performance on unseen data.

This strategy helps to prevent overfitting, where the model performs well on the training data but poorly on new, unseen data. By evaluating the model on the testing set, users can ensure that it generalises well to other datasets.

Model Selection

After preparing and engineering the features, users select the most appropriate machine learning model based on their specific needs. The platform supports various models, each with unique strengths:

- Linear Regression: Ideal for predicting continuous variables, such as overall quiz scores.
- Logistic Regression: Used for classification tasks, such as predicting whether a student's level in the next quiz.
- Decision Tree: Useful for easy-to-interpret classification tasks, offering a visual representation of decision rules.
- Random Forest: Combines multiple decision trees to enhance prediction accuracy and handle a larger variety of data patterns.
- Neural Networks: Highly versatile, used for capturing complex patterns in data for both regression and classification tasks.
- XGBoost: An advanced gradient boosting algorithm that is highly effective for both classification and regression tasks, known for its speed and performance. XGBoost can handle a wide range of data patterns and is often used in competitive machine learning due to its robust handling of large datasets and complex interactions.

The choice of model depends on the nature of the problem, the type of data, and the desired outcome. Users can experiment with different models to identify the best fit for their specific educational context.

Model Training

Once the data is prepared, the training process starts. The platform uses advanced machine learning algorithms to analyse the input data and learn the underlying patterns. During this phase, the model adjusts its internal parameters to minimise prediction errors. The training process involves several iterations, where the model continuously refines its predictions by comparing them against actual outcomes.

Model Evaluation

Once the model is trained, it undergoes evaluation using a separate validation dataset. This step assesses the model's performance by comparing its predictions with actual outcomes. Key metrics are calculated to determine how well the model is performing:

- Accuracy: The proportion of true results (both true positives and true negatives) among the total number of cases examined.
- Precision: The proportion of true positive results among all positive results predicted by the model.
- Recall: The proportion of true positive results among all actual positive cases.
- F1-Score: The harmonic mean of precision and recall, providing a single metric for model performance.
- Mean Squared Error (MSE): Measures the average squared difference between predicted and actual values, used for regression tasks.
- Root Mean Squared Error (RMSE): The square root of MSE, providing a measure of error magnitude.
- R-Squared (R^2): Indicates the proportion of the variance in the dependent variable that is predictable from the independent variables.
- Area Under the ROC Curve (AUC-ROC): Used for classification tasks, providing a measure of a model's ability to distinguish between classes.
- Confusion Matrix: A table used to evaluate the accuracy of a classification model by comparing predicted and actual values.

Users can review these metrics to understand the model's strengths and limitations, ensuring that the selected model meets their specific needs and performs well on the intended tasks.

Model Deployment

After evaluation, the model can be deployed to make predictions on new, unseen data. Educators can use these predictions to gain insights into student performance, identify at-risk students, and tailor instructional strategies accordingly. The custom model provides a dynamic and flexible tool for enhancing educational outcomes through data-driven decision-making.
By allowing users to select features, engineer data, and choose the most suitable model, the custom model training feature empowers educators to create highly specialised and effective predictive tools that cater to their unique educational environments.

Development of a Step-by-Step User Guide

A comprehensive Step-by-Step User Guide has been created for both educators and students, accessible from the homepage and the about page. This guide offers clear instructions on utilising the management tool effectively. Educators gain insights into analysing student data, tracking performance levels, and customising instruction to meet individual needs.

Challenges and Limitations

In a website utilising data analysis and AI within an educational context, several challenges and limitations are encountered, affecting various facets of functionality, usability, and performance. A fundamental challenge lies in the availability and quality of data. Educational datasets may be limited in size or scope, hindering the effectiveness of AI algorithms that rely on extensive and diverse data for training. To address this challenge, users are encouraged to actively share data and add participants to enrich the dataset, ensuring that AI models can learn from a broader range of examples and contexts.
Moreover, it's essential to recognize that AI in education is relatively recent, presenting numerous challenges in its implementation and adoption. One such challenge is the need to establish trust and understanding among educators and stakeholders regarding the role of AI. AI should not be perceived as making decisions autonomously but rather as a tool that offers insights and assists educators in repetitive tasks, such as grading or personalised learning recommendations. Educators must be actively involved in interpreting AI-generated insights and integrating them into their teaching practices.

Additionally, the interpretability of AI models poses a significant challenge in the education context. Complex algorithms may produce predictions or recommendations that are difficult to explain or understand, raising concerns about transparency and accountability. Educators may be hesitant to rely on AI-driven insights without a clear understanding of how they are generated and their potential impact on student learning outcomes.

Furthermore, navigating ethical and regulatory considerations surrounding data privacy and security is paramount in educational AI applications. Websites must adhere to stringent privacy regulations and implement robust security measures to protect sensitive student information from unauthorised access or misuse. Ensuring compliance with regulations such as GDPR requires careful attention to data handling practices and transparency in how AI algorithms process and utilise student data.

Despite these challenges, the integration of AI in education holds immense promise for improving teaching and learning outcomes. By addressing data limitations, fostering trust and understanding, promoting transparency and accountability, and prioritising ethical considerations, educational websites can leverage AI effectively to support educators and enhance student learning experiences. However, ongoing collaboration between educators, researchers, developers, and policymakers is essential to address emerging challenges and realise the full potential of AI in education.

Annexes

Additional Resources

Tools List

Programming Languages and Frameworks:

1. Python: A versatile programming language used for backend development, data analysis, and AI model implementation.
2. HTML (Hypertext Markup Language): The standard markup language for creating web pages and web applications.
3. JavaScript (JS): A programming language used for adding interactivity and dynamic content to web pages.

Data Analysis and Machine Learning Libraries:

4. Pandas: A powerful data manipulation and analysis library for Python, particularly useful for handling structured data such as tables and time series data.
5. NumPy: A fundamental package for scientific computing with Python, providing support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays.
6. Matplotlib: A plotting library for Python used to create static, animated, and interactive visualisations in Python.
7. Scikit-learn: A popular machine learning library for Python that provides simple and efficient tools for data mining and data analysis. It includes various algorithms for classification, regression, clustering, dimensionality reduction, and model selection.

Web Development Frameworks and Libraries:

8. Flask: A lightweight web application framework for Python used to build web applications, APIs, and microservices.
9. Jinja2: A modern and designer-friendly templating engine for Python used to generate dynamic web content, often used with Flask and Django frameworks.

Build and Deployment Tools:

10. Webpack / Babel: Build tools used to bundle JavaScript modules, optimise assets, and transpile modern JavaScript code into backward-compatible versions for browser compatibility.
11. Apache / Nginx: Web server software used to serve static content and reverse-proxy dynamic content over HTTP, commonly used to host web applications and APIs.
12. Gunicorn (Green Unicorn) / uWSGI: Application servers used to serve Python web applications, handling HTTP requests and managing multiple worker processes.

Asynchronous Task Management and Messaging:

13. Axios: A popular JavaScript library used to make HTTP requests from the browser, allowing for asynchronous communication between the frontend and backend of web applications.

Database Management and Cloud Services:

14. PostgreSQL (psql): A powerful open-source relational database management system used for storing and managing structured data, commonly used in web applications for data persistence.
15. Google Cloud Platform (GCP): A suite of cloud computing services offered by Google, providing infrastructure, platform, and software services for building, deploying, and managing applications and services on Google's infrastructure.

Translation and Internationalisation:

16. Babel: A JavaScript compiler used to translate ECMAScript 2015+ code into a backwards-compatible version of JavaScript that can be run in older browsers. Babel is commonly used in web development projects to support internationalisation and localization by enabling developers to write code in multiple languages and translate it into a single target language for deployment.

Emailing Service:

17. Emailing Service: An external service or library used for sending transactional emails, notifications, and newsletters from the web application. This service may include SMTP (Simple Mail Transfer Protocol) servers, third-party email service providers (e.g., SendGrid, Mailgun), or custom email delivery solutions integrated into the platform. Emailing services are essential for maintaining communication with users, sending account-related notifications, password resets, and other email-based interactions within the application.

These tools collectively form the technical foundation of the platform, enabling various functionalities such as data processing, machine learning, web development, frontend and backend operations, containerization, deployment, scalability, and database management.

Appendix A – Commonly Used Machine Learning Algorithms

Algorithm	Category	Description	Common Applications	Key Advantages	Key Disadvantages
Linear Regression	Supervised Learning	Models the relationship between a dependent variable and one or more independent variables using a linear approach.	Predictive analysis, risk assessment, time series forecasting.	Simple to implement and interpret, computationally efficient.	Assumes linearity, sensitive to outliers, may not capture complex patterns.
Logistic Regression	Supervised Learning	Used for binary classification problems, modelling the probability of a binary outcome based on one or more predictor variables.	Binary classification tasks like spam detection, medical diagnosis.	Easy to implement, interpret, and regularise; provides probability scores.	Assumes linearity in the logit, can be less powerful with non-linear data.

Decision Trees	Supervised Learning	Non-linear classification and regression algorithm that splits the data into subsets based on feature values, creating a tree structure.	Customer segmentation, credit scoring, predictive maintenance.	Easy to understand and interpret, handles both numerical and categorical data.	Prone to overfitting, sensitive to noisy data, can create complex trees.
Random Forest	Supervised Learning	An ensemble method using multiple decision trees to improve robustness and accuracy by averaging multiple tree predictions.	Fraud detection, stock market analysis, image recognition.	Reduces overfitting, handles large datasets well, and provides feature importance.	Computationally intensive, less interpretable than single decision trees.
Support Vector Machine (SVM)	Supervised Learning	Finds the hyperplane that best separates the data into classes, often used for classification tasks.	Text classification, image recognition, bioinformatics.	Effective in high-dimensional spaces, works well with clear margin of separation.	Not suitable for large datasets, less effective with overlapping classes.
K-Nearest Neighbors (KNN)	Supervised Learning	Classifies a data point based on the majority class among its k nearest neighbours in the feature space.	Handwriting recognition, recommendation systems, anomaly detection.	Simple and intuitive, no training phase, works well with small datasets.	Computationally expensive during prediction, sensitive to irrelevant features.
Naive Bayes	Supervised Learning	Based on Bayes' theorem, this probabilistic classifier assumes independence among predictors.	Spam filtering, sentiment analysis, document classification.	Simple and fast, works well with high-dimensional data, handles missing data well.	Assumes feature independence, can be less accurate with correlated features.
K-Means Clustering	Unsupervised Learning	Partitions data into k clusters where each data point belongs to the cluster with the nearest mean.	Market segmentation, document clustering, image compression.	Simple to implement, scalable, works well with large datasets.	Requires pre-specification of k, sensitive to initial placement and outliers.
Principal Component Analysis (PCA)	Unsupervised Learning	A dimensionality reduction technique that transforms data into a set of orthogonal components, preserving most of the variance.	Feature reduction, noise reduction, data visualisation.	Reduces complexity, helps in visualisation, retains essential data variance.	May lose important information, components are often not easily interpretable.
Neural Networks	Supervised Learning	Consists of layers of interconnected nodes (neurons) that can model complex patterns in data, particularly useful for large datasets.	Image and speech recognition, natural language processing, game playing.	Capable of capturing complex relationships, adaptable to various types of data.	Requires large amounts of data and computational resources, prone to overfitting.
Gradient Boosting Machines (GBM)	Supervised Learning	An ensemble technique that builds models sequentially, each new model correcting errors of the previous ones.	Web search ranking, predictive modelling, recommendation systems.	High predictive accuracy, robust to overfitting, works well with structured data.	Computationally expensive, can be slow to train, sensitive to hyperparameters.
Recurrent Neural Networks (RNNs)	Supervised Learning	A class of neural networks where connections between nodes form a directed graph	Time series forecasting, language modelling,	Capable of modelling temporal dynamics,	Prone to vanishing gradient problems, it can be difficult to train.

		along a sequence, useful for sequential data.	speech recognition.	effective with sequential data.	
Linear Regression	Supervised Learning	Models the relationship between a dependent variable and one or more independent variables using a linear approach.	Predictive analysis, risk assessment, time series forecasting.	Simple to implement and interpret, computationally efficient.	Assumes linearity, sensitive to outliers, may not capture complex patterns.
Logistic Regression	Supervised Learning	Used for binary classification problems, modelling the probability of a binary outcome based on one or more predictor variables.	Binary classification tasks like spam detection, medical diagnosis.	Easy to implement, interpret, and regularise; provides probability scores.	Assumes linearity in the logit, can be less powerful with non-linear data.

References

- Holmes, W., Bialik, M., & Fadel, C. (2019). Artificial intelligence in education: Promises and implications for teaching and learning. Center for Curriculum Redesign.
- UNESCO (2021). Artificial intelligence and education: Guidance for policy-makers.

Glossary of Terms

Algorithm: A set of rules or instructions given to an AI or a computer program to help it learn on its own from data patterns.
Apache: An open-source web server software used to serve static content and reverse-proxy dynamic content over HTTP, commonly used to host web applications and APIs.
Application Server: Software that provides an environment where applications can run, regardless of what the applications are or what they do. Examples include Gunicorn and uWSGI.
Asynchronous Task Management: The process of managing tasks that run asynchronously, allowing for operations to be performed without blocking the main process. Often used in web development for handling HTTP requests and background tasks.
Axios: A popular JavaScript library used to make HTTP requests from the browser, allowing for asynchronous communication between the frontend and backend of web applications.
Babel (JavaScript): A JavaScript compiler used to translate ECMAScript 2015+ code into a backwards-compatible version of JavaScript that can be run in older browsers.
Babel (Email Service): An external service or library used for sending transactional emails, notifications, and newsletters from the web application.
Data Analysis: The process of inspecting, cleansing, transforming, and modeling data with the goal of discovering useful information, informing conclusions, and supporting decision-making.
Decision Trees: A non-linear classification and regression algorithm that splits the data into subsets based on feature values, creating a tree structure.
Flask: A lightweight web application framework for Python used to build web applications, APIs, and microservices.
Google Cloud Platform (GCP): A suite of cloud computing services offered by Google, providing infrastructure, platform, and software services for building, deploying, and managing applications and services on Google's infrastructure.
Gradient Boosting Machines (GBM): An ensemble technique that builds models sequentially, with each new model correcting errors of the previous ones.

Gunicorn (Green Unicorn): A Python WSGI HTTP server for UNIX, used to serve Python web applications, handling HTTP requests and managing multiple worker processes.
HTML (Hypertext Markup Language): The standard markup language for creating web pages and web applications.
Jinja2: A modern and designer-friendly templating engine for Python used to generate dynamic web content, often used with Flask and Django frameworks.
JavaScript (JS): A programming language used for adding interactivity and dynamic content to web pages.
K-Means Clustering: An unsupervised learning algorithm that partitions data into k clusters where each data point belongs to the cluster with the nearest mean.
K-Nearest Neighbors (KNN): A supervised learning algorithm that classifies a data point based on the majority class among its k nearest neighbours in the feature space.
Linear Regression: A supervised learning algorithm that models the relationship between a dependent variable and one or more independent variables using a linear approach.
Logistic Regression: A supervised learning algorithm used for binary classification problems, modelling the probability of a binary outcome based on one or more predictor variables.
Machine Learning: A branch of artificial intelligence that involves training algorithms to learn from and make predictions or decisions based on data.
Matplotlib: A plotting library for Python used to create static, animated, and interactive visualisations in Python.
Neural Networks: Consists of layers of interconnected nodes (neurons) that can model complex patterns in data, particularly useful for large datasets.
Nginx: Web server software used to serve static content and reverse-proxy dynamic content over HTTP, commonly used to host web applications and APIs.
NumPy: A fundamental package for scientific computing with Python, providing support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays.
Pandas: A powerful data manipulation and analysis library for Python, particularly useful for handling structured data such as tables and time series data.
PostgreSQL (psql): A powerful open-source relational database management system used for storing and managing structured data, commonly used in web applications for data persistence.
Principal Component Analysis (PCA): An unsupervised learning algorithm that transforms data into a set of orthogonal components, preserving most of the variance, used for dimensionality reduction.
Python: A versatile programming language used for backend development, data analysis, and AI model implementation.
Random Forest: A supervised learning algorithm and ensemble method using multiple decision trees to improve robustness and accuracy by averaging multiple tree predictions.
Recurrent Neural Networks (RNNs): A class of neural networks where connections between nodes form a directed graph along a sequence, useful for sequential data.
Scikit-learn: A popular machine learning library for Python that provides simple and efficient tools for data mining and data analysis. It includes various algorithms for classification, regression, clustering, dimensionality reduction, and model selection.
SHAP Values: SHAP (SHapley Additive exPlanations) values are a method used in machine learning to explain the output of a model by attributing the importance of each feature to the model's prediction. SHAP values provide insights into how each feature contributes to the model's output for a specific instance, offering a comprehensive understanding of the model's decision-making process.
Supervised Learning: A type of machine learning where the model is trained on labelled data, meaning each training example is paired with an output label. Examples include Linear Regression, Logistic Regression, Decision Trees, Random Forest, SVM, KNN, Naive Bayes, Neural Networks, GBM, and RNNs.
Support Vector Machine (SVM): A supervised learning algorithm that finds the hyperplane that best separates the data into classes, often used for classification tasks.

Unsupervised Learning: A type of machine learning where the model is trained on unlabeled data and must find patterns and relationships within the data. Examples include K-Means Clustering and PCA.
Web Development Framework: A software framework designed to support the development of web applications, including web services, web resources, and web APIs. Examples include Flask.