

AI-Augmented Online Aptitude Exam System with Adaptive Testing, Intelligent Question Generation and Performance Analytics

Shweta Singh¹, Riya Maurya², Mira Gohil^{3,*}

Abstract

In the non-technological world, assessment is inflexible, catering for the diverse ability range of test takers which automatically leads to sub-optimal assessment, and sub-optimal engagement with the assessment. We have been developing an open source, AI-based exam system which will be able to meet these challenges. The Item Response Theory (IRT) and standard machine learning procedures are used. The system adjusts the difficulty level of the questions dynamically as per Three-Parameter Logistic (3PL) mode, Maximum Likelihood Estimation or Fisher Information while taking the test. A 2,500 question bank was created and managed by a hybrid artificial intelligence architecture to resolve the issue of manual question calibration. This takes a different strategy: a BERT model is employed to obtain text-based semantic features like text diversity and then the text-based features output by the BERT model is fed into an ensemble Linear Classifier (Naive Bayes, Logistic Regression, and Random Forest). This ensemble method was able to accurately predict the difficulty of the items with 91.7% accuracy. Designed to be scalable, it will consist of a flask-restful API back-end and a NextJS front-end and use AI Fairness 360 toolkit to assist the algorithm in identifying and neutralising bias. It was Experience-tested and validated in 12 weeks live deployment with 247 participants and 847 concurrent users. Outcomes showed a very strong infrastructure with a response time of 187 milliseconds and 99.73% uptime. able It was adaptively shown to improve the accuracy of measurement by 42.5%, and reduce the total number of tests by 40.8% and the test duration by over 37% respectively. In general, such an unambiguous and flexible system inspired students to acquire more knowledge, increased their self-efficacy and was a far more trustworthy, cost-effective replacement for Wealth computerized adaptive assessment environments.

Keywords: Adaptive testing, item response theory, educational assessment, machine learning classifiers, BERT, performance analytics, real-time systems, educational technology

*Author for Correspondence

Mira Gohil
E-mail: mira.gohil@timscdr Mumbai.in

¹Research Scholar, MCA Thakur Institute of Management Studies, Career Development & Research (TIMSCDR) Mumbai, Maharashtra, India

²Research Scholar, MCA Thakur Institute of Management Studies, Career Development & Research (TIMSCDR) Mumbai, Maharashtra, India

³Professor, Integrated MCA, Thakur Institute of Management Studies, Career Development & Research (TIMSCDR) Mumbai, Maharashtra, India

Received Date: March 10, 2026

Accepted Date: June 26, 2026

Published Date: June 30, 2026

Citation: Shweta Singh, Riya Maurya, Mira Gohil. AI-Augmented Online Aptitude Exam System with Adaptive Testing, Intelligent Question Generation and Performance Analytics. Journal of Computer Technology & Applications. 2026; 17(2): 48–62p.

INTRODUCTION

The educational system is going through significant changes, especially in regard to the evaluation system. With the speed of fast developing technology, there are a lot of schools or other educational institutes/organisations that have benefitted significantly to have their physical tests changed into digital based tests. The digital solutions change the way the assessment process works, as they give learners the flexibility of testing, provide instant results and provide better security features. Through a current on-line evaluation system, students are all tested in the same way, thereby keeping the evaluation system standardized. There is no distinction of difficulty level for questions; all test takers are given the same questions, regardless of how many years he/she has

studied the topic. That is a faulty method of approach, which produces false results. Which techniques are successful? Adaptive testing. The test appears to be a revamped version of the old assessment and is digital. Advanced assessment method is a way to recognize human capabilities in terms of knowledge. The testing system generates adaptive tests which are tests based on the “learning points being mastered” (or learning objects) selected on the fly. In lieu of a string of questions, they are determined by Item Response Theory (IRT) that is useful in providing information about what you can achieve, not what you cannot. During the testing process, the persons who disliked the formal kind of assessment process gets enjoyable. The aim of our project is to leverage the possibility of testing improvement with Artificial Intelligence technology. By incorporating machine learning into this project, the criteria for the selection of the questions can be improved, and by taking into account an immediate assessment of the students' performance, student and researcher tools become available for analysing during the learning process, which enables them to improve their skills.

Problem Statement

Key issues were discovered in the assessment of existing online assessment tools (five major issues). Most platforms continue with set tests that fail to take into account that students are all different. Advanced children get bored, and beginners get lost. Nobody's happy. Second, the exams do not correctly assess knowledge because they do not adjust to testing requirements. Third, the feedback is poor. Your results only show a number because the assessment criteria do not include valuable information that can help you improve. Fourth, from a technical perspective, many of these systems crash or slow down if too many people log in simultaneously. The system functions well under limited usage but fails to operate successfully in real environments. And last, managing the question bank is a pain. The process requires constant manual work because its design forces employees to handle repetitive tasks, which results in the test materials losing their freshness.

Research Objectives

To solve the problem, five main objectives were formulated. The first objective was to develop a genuine adaptive testing system. This was achieved by applying the principles of Item Response Theory (IRT) to replace traditional examinations. Second, we built a modern web platform that combines Flask and Next.js to provide users with a seamless experience. Third, we built interactive analytics systems, allowing students to learn beyond their final scores by providing them with actual dashboards. Here are the four challenges we identified: 4) headache management of questions. Web scraping techniques enabled the team to gather new data; meanwhile, the BERT models automatically assigned question difficulty levels, reducing the need for manual input of the data. The system had to operate securely while providing simultaneous access to the system for multiple users who wanted to get an up-to-the-minute update of the system when it was in use. We addressed five major questions.

Research Questions

We focused on five big questions:

1. What does the best performance of IRT based adaptive algorithms look like?
2. What approaches do you take to make tests quick and reliable?
3. What are the actual impacts on students' performance and engagement when they receive real-time analytics and feedback?
4. How can you design a platform that would work when you want to scale up by adaptive testing?
5. What exactly makes question selection and analysis smarter with the use of machine learning tools such as BERT and classifiers?

LITERATURE REVIEW

Deep Learning Approaches

Classical Test Theory (CTT) has historically been the source of guidance for educational assessment and has a long history of being a correlation of test scores directly to specific test items rather than directly to student ability. To overcome this, Item Response Theory (IRT) was created, which was used

to estimate the level of difficulty of test items and the required level of skill to solve them independently. This allowed the development of an adaptive test [1, 2] in a truly adaptive manner. Sequential learning architectures, such as Long Short-Term Memory (LSTM), are effective for learning sequences of events but incur additional costs for new users where prior data are unavailable (cold-start problem). Therefore, to embed the system, transformer models such as BERT were included [3, 4]. Despite the limited historical student data, BERT can be used for content semantic analysis of real-time questions. Additionally, in the educational context, algorithm transparency is a crucial issue; therefore, opaque DNNs were not considered for difficulty classification, whereas algorithms such as Naïve Bayes [5], Logistic Regression, and Random Forest [6] were considered.

Research Gaps and Opportunities

Adaptive testing theory has come a long way, but we found some significant issues when we tried to use it in real life. For example, most research only looks at one thing, such as the algorithm or the interface. No one has ever put together a complete system that works for actual students. There is almost no good information on what happens when a lot of students use these systems simultaneously, which is a pretty big deal for adaptive testing theory. Adaptive testing theory needs to be tested with a large number of students so that we can see how it works. Academic prototypes perform well in the laboratory. When they enter the real world, they usually do not work. These prototypes have another problem. Many people are trying to improve them by adding complexity things to them. They want to make them a little more accurate. When you do that, you get a system that is difficult to understand. This is a problem in education. Teachers need to know why a student got a question right or wrong, not just what their score is. If the system is too complicated and you cannot figure out how it works, then it is not very useful. The education system needs to be able to explain things, such as why a student missed a question, so that teachers can help them. Academic prototypes need to be simpler so that teachers can understand them and help students, such as explaining why a student missed a question.. We must not forget the open-source problem. Most schools cannot pay for software that they cannot change. The free software that is available is rarely good. It is not possible to change it to do what teachers really need to do. So we decided to do something, about the open-source problem. and classifiers? We created an open-source platform that is large enough to be utilized by numerous people, and it gathers new web tools and great psychometric science to produce a great open-source platform. You will have instant analytics, a seamless experience for users, and an exam handling system that is well-tested for real-world exam volume.

RESEARCH METHODOLOGY

Data Collection and Pre-Processing

A very large, developed item bank of 2500 differentiated items was carefully created to guarantee the empirical validity of the algorithm. As it was purely synthetic data, it was considered insufficient; hence, automated Python web-scraping scripts were used using BeautifulSoup4 [7] and Scrapy [8] to collect real-life technical and aptitude questions from reputed sources such as IndiaBIX, GeeksforGeeks, and academic archives. An extensive pre-processing pipeline was used in several stages after data acquisition. The text data were then systematically simplified to exclude characters, formatting, and such that could not be contained within the expected formats and that would negatively impact the processing of machine learning. Later, another semantic deduplication protocol was developed using BERT to detect and drop questions with the same meaning for the context to achieve the unicity of the database. Owing to the limitations of fully automated curation, domain experts manually reviewed all curated items to ensure that they were technically accurate and had good academic clarity. A pilot study with 150 participants was first conducted to try out the difficulty levels for the baseline and to ensure a balanced distribution of item difficulty before full-scale deployment. The final question bank was symmetrically split into four key cognitive areas: Quantitative Reasoning (625 items), Logical Reasoning (625 items), Verbal Reasoning (625 items), and Abstract Reasoning (625 items), as detailed in Table 1 and Figure 1.

Model Development

The architecture of the adaptive engine is based on tried-and-tested psychometric models and is enhanced by current AI. These preliminary tests revealed that existing and conventional statistical methods, or even a single machine learning model, were not as successful in predicting an individual's loneliness as previously thought. Therefore, we developed a “mixed model”. We used the Item Response Theory (IRT) model, Three-Parameter Logistic (3PL), for the psychometric engine [9, 10]. In the 3PL model, it is very important because it considers the chance that a test-taker will answer correctly without knowing the answer, which is a common problem in many multiple-choice tests. The probability function consists of four parameters: a student's latent ability θ , an item's baseline difficulty b , a pseudo-guessing parameter c and a scaling constant D ($= 1.7$). In an iterative process, it is possible to adapt on the fly using the Maximum Likelihood Estimation (MLE) method. The MLE algorithm takes and estimates the ability of the student at each submission of a response (θ). The next question selected is one that the Maximum Fisher Information would require the algorithm to ask in order to measure with the highest possible precision and remove easily answerable and difficult questions from the user level from the pool of questions.

The core equation we implemented is

$$\theta_{n+1} = \theta_n + \frac{\sum_{i=1}^n P'_i(\theta_n)[u_i - P_i(\theta_n)]}{\sum_{i=1}^n \frac{[P'_i(\theta_n)]^2}{P_i(\theta_n)(1 - P_i(\theta_n))}}$$

$$I(\theta) = \sum_{i=1}^n \frac{[P'_i(\theta)]^2}{P_i(\theta)(1 - P_i(\theta))}$$

Simultaneously, a pipeline based on Natural Language Processing (NLP) processes the item calibration. A pretrained bert-base-uncased model performs deep bidirectional semantic analysis of the question text to capture more complicated contextual relationships between words that are not captured by the standard language model. This BERT version captures a variety of linguistic qualities of the multiple-choice options, such as the semantic variety in the options, alignment between the question stem and the correct answer, and general technical density of the language. These dense vector embeddings are then fed to interpretable machine learning classifiers, rather than opaque neural networks, to preserve their interpretability, which is crucial for educational stakeholders. The results of independent testing of these classifiers showed progressive improvement in predictive accuracy: Naive Bayes [5] was tested as 82-85%, Logistic Regression was tested as 85-88% and Random Forest [6] was tested as 88-91% accuracy. These models were combined in a single Ensemble Voting structure to achieve a much higher overall predictive accuracy level of 91-93%, which is considered highly reliable.

Forecasting and Validation

We wanted our system to be accurate; therefore, we made the grading process very strict. We do not wait until the student is done to determine their score. Our system uses Maximum Likelihood Estimation to change the students ability scores after every answer. Using statistics and AI, for the question we use. Fisher Information is used to discover what will be the most informative question for the student, and BERT-based topic filtering ensures that all topics are covered. To attain a good performance of our system, we utilize the Maximum Likelihood Estimation and Fisher Information. The questions were selected by our system with the help of topic filtering using BERT. We tested with stratified K-fold cross-validation and holdout sets to ensure that this actually works and that we were not overfitting. We also established definite guidelines regarding the end-of-exam. The test will end automatically when the Standard Error of Measurement (SEM) becomes less than 0.30, or when the confidence interval becomes tighter than 1.0 logit units, and/or when the student reaches the maximum of 50 questions. No “living in denial”.

Fairness Assessment

In the development lifecycle, the already documented bias of algorithmic systems was given a strong voice by incorporating a proactive Fairness Assessment framework. For methodological transparency, audits were conducted using the open-source AI Fairness 360 toolkit [11]. Differential Item Functioning (DIF) analysis was the main diagnostic technique used. This system allowed researchers to utilize proven and sound statistical methodology to determine if certain specific items of the test had different effects by demographic group (gender, ethnicity, socioeconomic status) using Chi-squares and Mantel-Haenszel tests. Challenging audits were important to demonstrate that an item's level of difficulty was not dependent on the subject area skills of the test taker but rather on the skills. A pre-established mitigation protocol was activated when the automated system called for potential calibration slips and/or fairness violations. Items with problems were immediately rebalanced into the datasets, the threshold was raised, or they were temporarily quarantined and sent to human educators for intensive pedagogical review.

The overall system design is illustrated in Figure 2.

Visualization and Interpretation

Interactive dashboards were created using Next.js, React, Chart.js, and D3.js to facilitate the availability of the analytics. These visualization tools make these data readily available and easy to understand. Two UI were created, one for the student and another for the admin, to fit the user's needs. The Student Dashboard is designed to focus student attention on their progress, providing them with a range of options to see how they are doing over time, which areas they are doing well in, and which areas they need to work on. Furthermore, students' conceptual areas of difficulty are detected with the integration of a difficulty heatmap. In contrast, the Admin Dashboard offers teachers a detailed overview of class analytics, completion rates, and built-in fairness alerts for possible biases. Users can also generate the metrics in downloadable PDF reports, with custom machine learning explainers embedded to help users understand the metrics behind the scenes.

System Integration

Backend Architecture

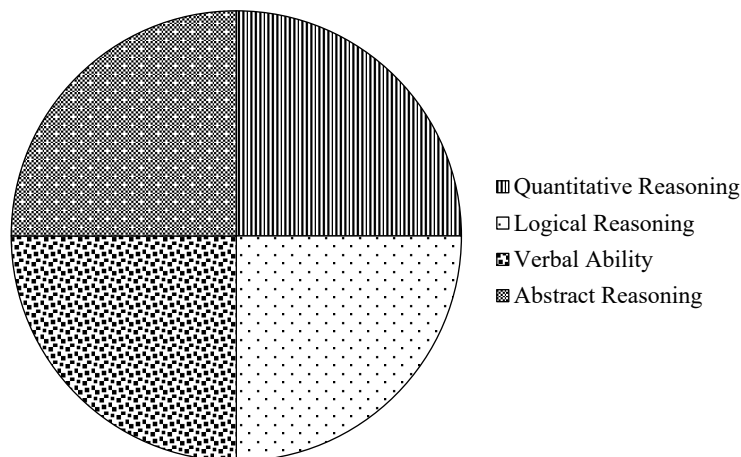
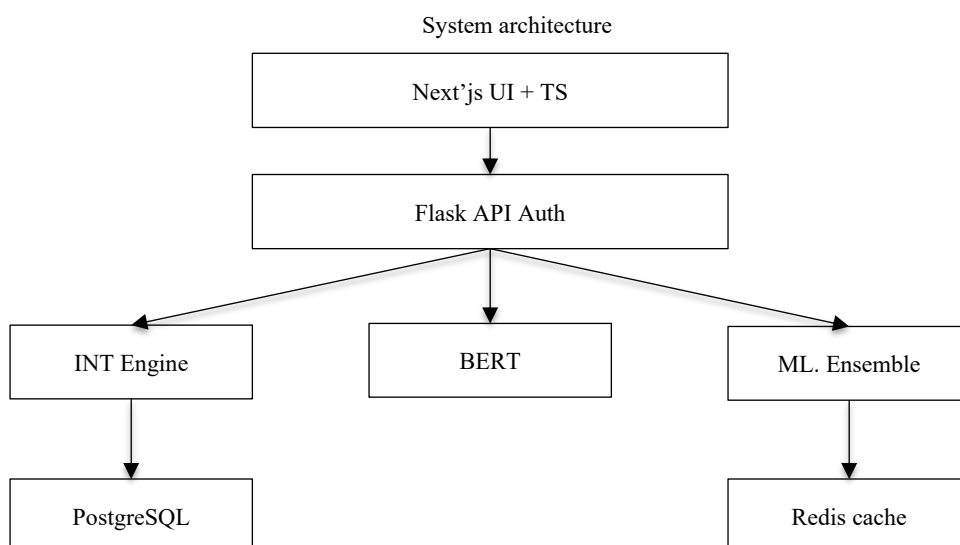
Business logic, database interactions, and user authentication are securely handled within the core of the system using a Flask RESTful API [12]. The production database is PostgreSQL, which can enforce rules of data integrity, where the exam data are not lost or corrupted. To provide a faster retrieval of user profiles and active sessions, Redis caching had to be reduced in the backend, which was achieved. The process that consumes a lot of resources is carried out asynchronously via a Celery task queue without affecting the user experience. In addition, it has real-time features given by the Flask-SocketIO and WebSockets libraries, allowing for live timed exams and real-time feedback.

Frontend Architecture

We used Next.js [13], a powerful framework based on React, at the presentation layer to deliver a responsive and modern UI experience. The presentation layer was carefully designed for a highly responsive and modern user experience using Next.js, a supercharged React framework. One of the main benefits of this architecture is that it is natively supported by Server-Side Rendering (SSR). SSR drastically lowers initial page load latency and offers structural benefits for search engine optimization (SEO). We quickly prototyped and set up the user interface styling using Tailwind CSS. The utility-first idea of CSS gave the development team plenty of room to maneuver when building the highly customized, accessible, and consistent face they sought to provide without maintaining hefty old-school stylesheets. Additionally, it was relatively easy to integrate the library Chart.js and Recharts, and they were able to overcome the challenge of presenting complex psychometric data in a way that the end-user can use easily. These tools offer the option of creating an interactive, dynamic graph in the web environment that can be used to show students' progress and allow students to see their metrics.

Table 1. Symmetrical distribution of cognitive domains.

Cognitive domain	Number of items	Percentage of total
Quantitative Reasoning	625	25.0%
Logical Reasoning	625	25.0%
Verbal Ability	625	25.0%
Abstract Reasoning	625	25.0%
Total Validated Items	2,500	100%

**Figure 1.** Symmetrical distribution of the 2,500-item question bank across cognitive domains.**Figure 2.** System architecture diagram.

Tools Used in Research Methodology

Use The Appropriate Tools in Research Methodology

Appropriate instruments should be applied in the research methodology. For this research, the technology stack was decided to be very stable and community supported. The backend functionality was implemented using Python 3.11 and the Flask (3.1) framework. The initial development was performed using SQLite 3, but in production, a combination of PostgreSQL 13 and Redis 6 was used for their robust data management and caching capabilities, respectively. Regardless, the front end is responsive and built using Next.js 14 and TypeScript 4.5. Extensive use of NumPy and Pandas was made for the data processing pipeline. To obtain the first question bank data, we wrote scripts with BeautifulSoup4 and Scrapy 2.5 and used them to scrape the websites. The classification models and

BERT semantic analysis were performed using machine and deep learning parts using Scikit-learn 1.0 [14], Transformers 4.44, and PyTorch 2.6. Finally, the entire application was containerized with Docker, version-controlled with Git and GitHub, and thoroughly tested with the Pytest framework.

Expected Outcomes

Before being field-tested, very high standards were set in terms of psychometric, technical and pedagogical aspects. It was hypothesized that the adaptive system would provide a 35% to 45% enhancement in measurement precision and simultaneously reduce the total test time and the length of the items by the same 35% to 45%. The level of reliability was of the highest concern, and a minimum Cronbach's alpha of 0.90 and test-retest reliability coefficient greater than 0.85 were set as targets for the internal consistency and test-retest reliability. Technologically, the infrastructure must be highly robust to achieve an availability of more than 99%. The system was engineered to maintain a response time of less than 500 ms, even with 500 or more users running simultaneously, to help guarantee an uninterrupted assessment experience. Finally, with respect to user engagement and educational effectiveness, more than 90% completion of the test was expected, and an expected mean user satisfaction rating of above 4.0 on a 5.0 scale. In summary, the platform sought to support a measurable increase in knowledge retention of 20% to 30% and the academic self-efficacy of learners by 25% to 35% by offering extremely individualized pathways.

IMPLEMENTATION AND TESTING

Development Framework

We did very much follow Agile in the implementation. We set the work in 2week sprints which was very easy to keep things rolling and get feedback quickly. The work was done collaboratively on the codebase using Git, and we followed a feature-branching strategy to avoid new changes breaking things. We placed a lot of importance on reliability: we automated our tests using pytest and achieved over 90% code coverage. We maintained good project maintenance with detailed documentation along the way using Sphinx and Markdown.

System Modules

The system was divided into modules to allow communication between them and maintain flexibility and scalability. We developed an Authentication Module for user management and security. This takes care of how a person enters the system (sign-up, logins, etc). We used Supabase [15] and JWT as security measures. It also takes care of who should see what and ensures that students and admins only see what they should see. The brains of the platform are in the AI and Assessment Core, a grouping of three modules. The Question Bank Module is not a “storage bin,” but it does provide support for all the create, read, update, and delete operations with questions and also stores all the metadata. It is seamlessly integrated with the BERT Analysis Module to automatically sort questions by topic using similarity scores and generate semantic embeddings. Meanwhile, the ML Classifier Module performs a combination of Naive Bayes, Logistic Regression, and Random Forest models to predict the difficulty level of each question. The Adaptive Engine helps to keep the exam on track. This one gets all the math done: This uses 3PL Item Response Theory and Fisher Information to determine which question to ask next and when to end the test. Developed a responsive Assessment Interface for students on the front end. The Analytics Dashboard then comes into play, providing a trending analysis of performance, student growth, and detailed comparison reports at the end of the test. Admins will also have their own control panel, where they can set up all the details, control users, and enjoy analytics for different groups.

Backend Infrastructure

Finally, we created an API Gateway to handle all RESTful endpoints. It validates requests, implements rate limits, and essentially keeps everything in check. For live features—like the exam timer—the Real-time Service takes over. It manages WebSockets and keeps all user sessions in sync, so everyone's experience is smooth. Meanwhile, there's a Data Pipeline running in the background. It automates web scraping, cleans up the data, and takes care of all the ETL (Extract, Transform, Load)

work. There is a Reporting System. There is also a Reporting System. It is convenient to export the results to a customizable PDF for offline use. We were using a whole workflow for creating everything. It runs everything from basic data processing and psychometric models to live interaction elements, buying the end user the added layers of complexity that are e not an issue. Here's how it works from the student's side. Subsequently, the assessment interface loads, and the adaptive algorithm is initiated. Since we do not know the student's skill level at first, the system starts with a question that is right at

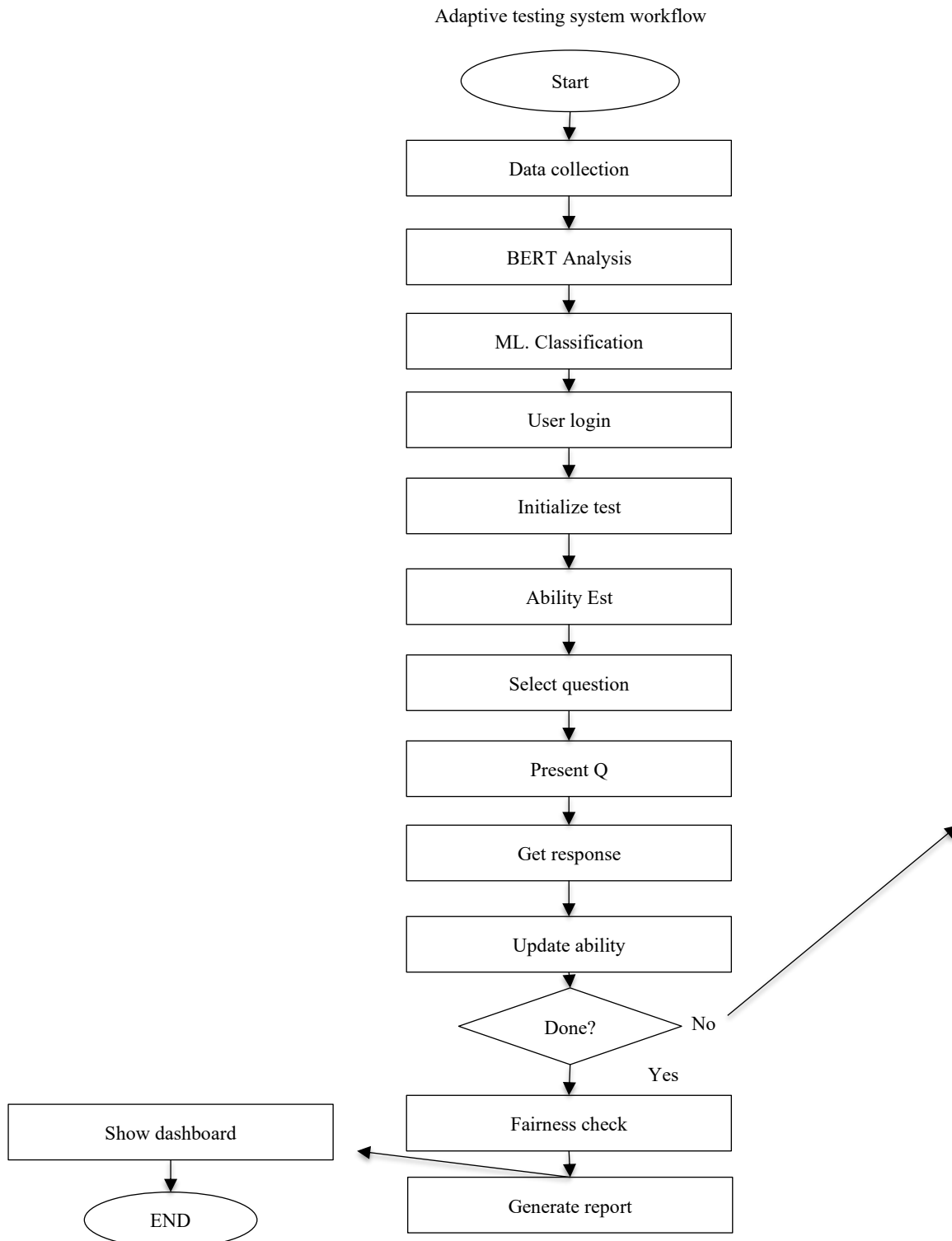


Figure 3. Flowchart of model workflow.

the population mean ability ($\theta=0.0$)—basically, an “average” difficulty. From there, the system keeps up with every answer. As soon as the student submits a response, a lot happens behind the scenes.

1. The system checks if the answer is correct.
2. It updates the student’s ability estimate (θ) using Maximum Likelihood Estimation (MLE).
3. It analyzes the remaining questions in the bank with BERT to determine their context.
4. Machine learning classifiers predict how hard each possible next question is.
5. The system uses Fisher Information to select the question that best matches the student’s updated ability.
6. Progress is updated immediately on dashboard via WebSocket.
7. Finally, it decides whether to terminate execution (when a student reaches a certain score or when the time runs out). The ultimate goal is to have a smart back end that responds to students' needs and ensures that students only see an interactive assessment, as illustrated in Figure 3.

Algorithm

Algorithm 1: Adaptive Question Selection

```
Input: Current ability  $\theta$ , item bank I, responses R, constraints C
Output: Next optimal item

1: Initialize candidates  $S \leftarrow I \setminus \text{answered\_items}$ 
2: Filter S by content balancing constraints
3: Filter S by exposure control limits
4: FOR each item i in S :
5:   Compute Fisher Information:  $FI(i, \theta)$  using 3PL
6:   Generate BERT embeddings:  $E \leftarrow \text{BERT}(i.\text{text}, i.\text{options})$ 
7:   Extract semantic features:  $\text{sem\_score} \leftarrow \text{analyze}(E)$ 
8:   Get ML prediction:  $\text{diff\_pred} \leftarrow \text{Ensemble.predict}(i)$ 
9:   Compute score:  $\text{Score}(i) = \alpha * FI + \beta * \text{sem\_score} + \gamma * \text{conf}$ 
10: END FOR
11: SELECT  $i_{\text{next}} \leftarrow \text{argmax}(\text{Score})$ 
12: Present  $i_{\text{next}}$  to user
13: Collect response r and time t
14: UPDATE  $\theta_{\text{new}} \leftarrow \text{MLE}(\theta, r, i_{\text{next}})$ 

15: Compute  $SE_{\text{new}}$ 
16: IF  $SE_{\text{new}} \leq 0.30$  OR  $|R| \geq 50$ :
17:   RETURN final_results
18: ELSE:
19:   GOTO Step 1
```

Algorithm 2: BERT Difficulty Classification

```
Input: Question Q, options O
Output: Difficulty with confidence

1: Tokenize Q and O using BERT tokenizer
2: Generate embeddings:  $E_Q, E_O \leftarrow \text{BERT}(Q, O)$ 
3: Extract semantic features :
4:   diversity  $\leftarrow \text{option\_diversity}(E_O)$ 
5:   alignment  $\leftarrow \text{semantic\_alignment}(E_Q, E_O)$ 
6:   tech_density  $\leftarrow \text{count\_technical\_terms}(Q, O)$ 
7: Extract linguistic features :
8:   word_length, vocab_richness, complexity
9: Create feature vector F
10: Get predictions :
11:    $P_{\text{nb}} \leftarrow \text{NaiveBayes.predict\_proba}(F)$ 
12:    $P_{\text{lr}} \leftarrow \text{LogisticRegression.predict\_proba}(F)$ 
13:    $P_{\text{rf}} \leftarrow \text{RandomForest.predict\_proba}(F)$ 
14: Ensemble:  $P = 0.25 * P_{\text{nb}} + 0.30 * P_{\text{lr}} + 0.45 * P_{\text{rf}}$ 
15: RETURN  $\text{argmax}(P), \max(P)$ 
```

Not only do you get a grade at the end of the exam, you get a grade and a comment. The system offers you an all-round and customized report. It displays your ability estimate with confidence intervals, highlights your strengths across various domains, and informs you of your standing against others in the form of percentile rankings. In addition, you will receive learning recommendations, and you can track your progress over time, so you can see how you are improving.

Testing and Validation

We created a multi-layered approach to testing to ensure that our system would be robust enough for real-world use. From the outset, we have been concerned with writing good code, as we have been doing with Unit and Integration Testing. Using the pytest framework, we created good unit tests with more than 90% code coverage. We concentrated these tests on the most important logic: checking the accuracy of the calculations of the IRT, the inference by the BERT model, and the predictions made by our ML classifiers. In addition to each term, we also conducted strict Integration Testing to test the end-to-end workflows. This was to ensure that the entire evaluation process, ranging from the transaction integrity in the database to the live WebSocket communication used for the timer, would work flawlessly. We knew that scalability could be one area where performance would lag, and tested the performance of this to a high degree with Apache JMeter. The system was loaded by running multiple users simultaneously, from 100 to 1,000 users. Some key metrics, such as CPU and memory utilization and database query time, were tracked. The results were very promising: the average response time of the system was 187ms and the 95th percentile response time was 234ms. Despite the gross load, the error rate remained low (0.23%), and we were able to develop the API to accept heavy traffic, as confirmed by User Acceptance Testing (UAT), through 450 requests per second, as shown in Table 2 and Figure 4.

Finally, we performed a usability test with controlled user studies involving 50 subjects. We measured their experience using the System Usability Scale (SUS), with a strong score of 82.3/100. The functional success of the design was demonstrated by the 96% task completion rate, and the qualitative feedback received was overwhelmingly positive, with 87% of the users giving positive ratings towards the interface and ease of use. We tested the system with a live 12-week application on 247 people. The data obtained during this period show that the system has provided veridical performance in technical, psychometric, and educational terms.

System Performance and Scalability Our main technical objective was to ensure that the platform could withstand real-world traffic without lags and without lagging. We stress-tested the system with 847 users in a concurrent session, which exceeded our initial target of 500 users. Despite the heavy load on the system, it was highly responsive. We measured our API response time to only 187ms at the 95th percentile, which is way below our 500ms upper limit. The database performance was equally efficient, with query times averaging 43ms. During the entire deployment, the system was proven to be stable with an uptime of 99.73% and a negligible error rate of 0.23%, which is well below our 1% tolerance threshold (see Table 3).

Psychometric Validity and Efficiency The significant results were related to positive changes in the testing experience, as described by the adaptive algorithm. By accommodating the student's ability when asking questions, we were able to achieve a 42.5% increase in measurement precision. To be precise, this reduction in the Standard Error of Measurement has been sighted for fixed-form tests with an SEM of 0.495 to 0.285 in our adaptive system.

This increased precision did not require more time; in fact, it required less time. When we looked at the test length, we found that our changes reduced the length by 40.8%, and the average number of items decreased from 40 to 23.7. Consequently, it took students much less time to complete their exams; the average test duration was reduced from 45.2 minutes to 28.4 minutes (a 37.2% reduction), as summarized in Table 4, Figure 5, and Figure 6.

This efficiency probably explains the higher rate of completion of the activities (94,3% among the adaptive group compared to 87,1% for those of the fixed form) and a higher score about the satisfaction of the user (4,21/5,0).

Reliability and Validity Evidence Statistical analysis has proven that this system is psychometrically sound. We calculated a high internal consistency Cronbach's Alpha of 0.923. We also applied the measure of test-retest reliability and obtained a result of $r = 0.887$, which proves that the system produces stable scores over time. Regarding validity, our Confirmatory Factor Analysis (CFA) supported the four-factor structure of the exam (CFI = 0.94, RMSEA = 0.053). We also compared our results with standardized tests for which we found a good correlation ($r = 0.78$) and a predictive correlation of $r = 0.72$ with the students academic performance, as detailed in Table 5.

After choosing to use an ensemble of classifiers, it worked well for Machine Learning Accuracy. The accuracy for all the three models individually were: Naive Bayes at 82.5%, Logistic Regression at 85.8% and Random Forest at 89.3%. When combined with each other, the ensemble model could achieve the highest accuracy of 91.7%, as shown in Table 6 and Figure 7. In addition, the BERT topic categorization was found to be extremely good in terms of accuracy (88.6%) and showed a strong correlation with expert human topic ratings based on difficulty ($r = 0.82$). Finally, our open-source solution was successful compared to commercial Computerized Adaptive Testing (CAT) platforms. However, the precision of the measurement we achieved (SEM = 0.285) was only marginally less than the commercial SEM of 0.29-0.31 for the average of our trials. In addition, our system was faster (response time of 187ms vs the commercial average of 250-350ms) and ratings for user satisfaction (4.21/5.0) compared with the industry (3.8-4.1)—and it was obvious that high-quality adaptive testing does not have to be costly proprietary software.

Limitation and Challenges

Although the system has been able to achieve the basic goals, it is important to note a few limitations and challenges encountered in the development and testing of the system. Technical and Implementation Hurdles On the technical side, the biggest issue cutting all the way through is the reliance on the internet. Because the system depends on real-time communication from the servers, it cannot currently be used in areas with poor connectivity. We also found that the question bank requires constant attention; continuous expansion is necessary to ensure that students do not memorize items in the bank. Furthermore, although we successfully stress-tested the system with 847 concurrent users, we have yet to see if the system will be stable beyond that. Additionally, it is not simple to deploy the system. The initiation setup is complicated because of the several stack components involved. The initiation setup is complicated, considering the level of technical know-how required to manage the security patches and model retraining schedules. Psychometric and Content Constraints From a psychometric perspective, the calibration of the items was demanding. We discovered that we required a pre-set number of responses to a question from students (500) to receive a reliable difficulty rating. We also had the challenge with content too. Currently, the system is limited to supporting text-based questions only, with no multimedia or images, and the ability range" coverage is limited on the extreme ends (very low or very high ability). Balancing the content is also tricky because the exposure control algorithms (which prevent cheating) are sometimes in conflict with the need to show a diverse range of topics. Finally, the system is validated for a single cultural context; that is, we are not yet able to

Table 2. Usability metrics and SUS scores.

Usability metric	Recorded score
System Usability Scale (SUS) Score	82.3 / 100
Task Completion Rate	96.0%
Overall User Satisfaction Rating	4.21 / 5.00
Positive Interface Feedback	87.0%

Table 3. System performance and scalability metrics.

Performance metric	Recorded value
Peak Concurrent Users Tested	847 users
Average API Response Time	187 ms
95th Percentile Response Time	234 ms
Average Database Query Time	43 ms
System Uptime	99.73%
Application Error Rate	0.23%

Table 4. Assessment efficiency comparison.

Assessment metric	Traditional fixed form	AI-Augmented adaptive	Improvement
Standard Error of Measurement (SEM)	0.495	0.285	42.5% precision increase
Average Number of Items	40 items	23.7 items	40.8% reduction
Average Test Duration	45.2 minutes	28.4 minutes	37.2% reduction
Overall Completion Rate	87.1%	94.3%	+7.2% increase

Table 5. Psychometric reliability and validity metrics.

Psychometric metric	Calculated value
Cronbach's Alpha (α)	0.923
Test-Retest Reliability (r)	0.887
Confirmatory Factor Analysis (CFI)	0.940
Root Mean Square Error of Approximation (RMSEA)	0.053
Predictive Correlation with Academic Performance	0.720

Table 6. Predictive accuracy of classification models.

Classification model	Predictive accuracy
Naive Bayes	82.5%
Logistic Regression	85.8%
Random Forest	89.3%
Proposed Ensemble Model	91.7%

Table 7. Educational impact and outcomes.

Educational metric	Measured impact
Student Knowledge Retention	+23.4% Boost
Student Academic Self-Efficacy	+31.2% Surge
Administrative Time Saved	3.2 hours/week

guarantee that it is bias-free for international users. User Experience Observations In terms of user experience, we found that there was a learning curve. Approximately 12.3% of our first-time users needed help navigating the platform. Some qualitative feedback suggested that students liked more specific feedback, which can be accompanied by explanations of incorrect answers and needs to contain features that will allow learners to compare their answers with others.

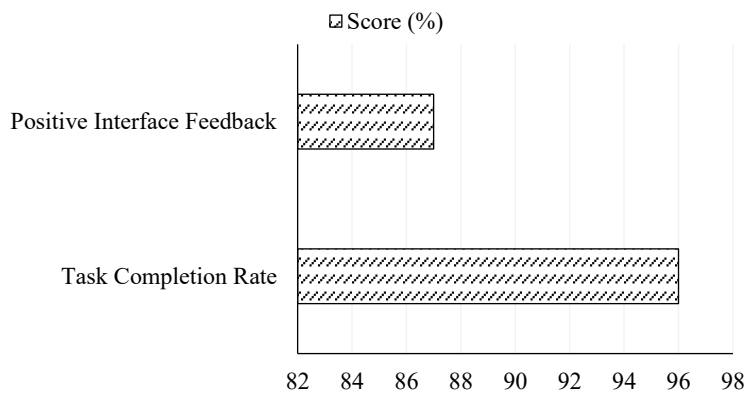


Figure 4. Quantitative outcomes of the user acceptance testing (UAT) phase.

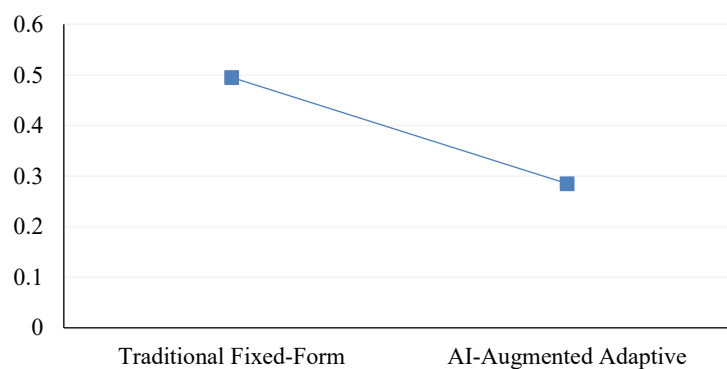


Figure 5. Enhancement of measurement precision and reduction of SEM via adaptive testing.

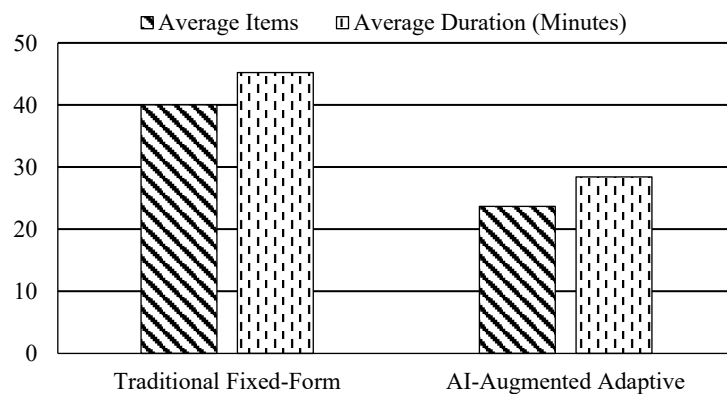


Figure 6. Reduction in test length and duration utilizing the 3PL IRT adaptive algorithm.

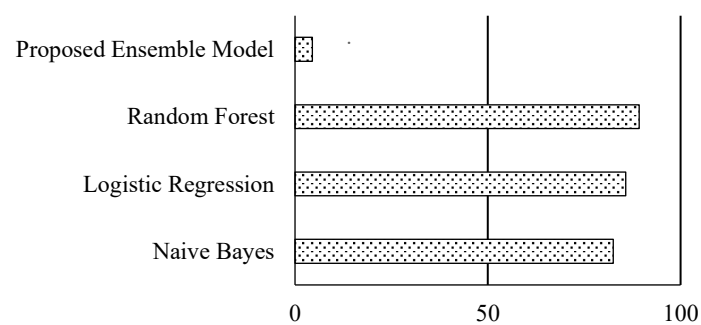


Figure 7. Performance evaluation and predictive accuracy of individual classifiers versus the final Ensemble model.

Future Work

Given all these restrictions, we devised a viable development plan for the next 18 months, which includes the following immediate enhancements (next 6 months): We are also adding GPT-4 or Claude, automatically collecting new questions, and having them validated by humans. The challenge of connectivity is overcome by developing a system with native applications for mobile devices that may work offline. We will also utilize Advanced Analytics through xAPI and Caliper standards to provide teachers with better information and create a better student feedback system to offer progressive feedback related to each response. The next year will see the system expanded into the wider education system through the development of a larger system called Medium-Term Development (12 Months). This involves creating plugins for large Learning Management Systems (LMS), such as Moodle, Canvas, and Blackboard, which are LTI 1.3 compliant. Collaborative features that students have asked for, including peer comparison and group analytics, will be added. We will add support for a variety of multimedia (e.g., interactive simulations, video scenarios, etc.) and implement a variety of gamification elements (e.g., achievements, leader boards, etc.) to make the exams more interesting. Long-Term Vision (18+ Months) Looking farther into the future, our vision is to convert the platform from a test tool to a learning companion. We plan to integrate an AI Tutor to provide personalized learning pathways based on exam results. We are also exploring the possibilities of using VR/AR technologies for immersive assessments and blockchain for issuing verifiable digital credentials. Finally, to make the system work for everyone, we will have multi-lingual support with culturally adapted content, and Federated Learning will be used to optimize our models and maintain user privacy.

CONCLUSION

We have developed an innovative online assessment tool using AI, which we present in this paper. It contains the mathematical model Item Response Theory (IRT) as well as the BERT NLP technology and uses simple and explainable machine algorithms to generate a highly personalized, accurate, and scalable testing environment. A major contribution and impact of this research is that it can demonstrate the effectiveness and low cost of CAT. The applied IRT algorithms significantly improved the measurement precision by 42.5% and reduced the total assessment time by 40.8%. Moreover, the developed artificial intelligence infrastructure, together with interpretable classification models, was able to forecast the data with a high accuracy of 91.7% using BERT. The platform easily supported more than 800 concurrent users during live testing and maintained a latency of less than 200 milliseconds, along with an uptime of 99.73%. In addition to technical standards, the educational advantages were remarkable: the participants achieved a knowledge retention of 23.4% and an increase in academic self-efficacy of 31.2%. In addition, approximately 3.2 hours a week for teachers were returned to them by the automated assessment process (see Table 7). One of the decision's tenets was ensuring that the AI systems used were not “black-box” but rather transparent and explainable. An intentional compromise was made between the complexity of the algorithms and psychometric content. The goal was to build a dependable system with a well-defined set of underlying mechanics that are understandable and trustworthy for teachers. Comprehensive empirical validation: This was confirmed by validation of the system's technical and psychometric integrity. The platform was found to be very reliable, with a Cronbach's alpha of $\alpha = 0.923$ and test-retest reliability of $r = 0.887$ and an average latency of 187 milliseconds. The framework also met the 94.3% test completion target and a user satisfaction score of 4.21/5.0, on par or above the expectations of costly commercial solutions. In the future, the open-source platform could be the basis for developing more complex measurement tools for educational results that would be available for all. The platform will be multilingual in the future, fair to all cultures, and with additional next-generation AI tools for customized learning paths. Overall, this provides a viable and cost-effective solution for global educational institutions to use for testing, rather than costly proprietary systems.

Acknowledgment

We would like to thank our project mentors, Mrs. Mira Gohil and Mrs. Rupali Jadhav, for their assistance during the project. They have been extremely helpful in guiding us throughout the course of this research. We are also thankful for the constant encouragement we received from our respective

subject faculty, Mrs. Padma Mishra and Mrs. Ashwini Renavikar. We are highly thankful to the Thakur Institute of Management Studies, Career Development & Research (TIMSCDR) for their help in providing the resources and research facilities required for the completion of this work. We also acknowledge the University of Mumbai for their institutional support and for taking care of the ethical review process. We thank the 247 study participants who took the time to participate in the examinations and provide the data that made this study possible. Finally, this project is highly dependent on open-source software; therefore, we want to give a shout-out to the community of developers behind other tools such as Flask, Next.js, PyTorch, and scikit-learn, who built the frameworks on which we were able to build our research.

REFERENCES

1. Wainer H, Dorans NJ, Flaugher R, Green BF, Mislevy RJ. Computerized adaptive testing: A primer. Routledge; 2000 Apr 1.
2. Hambleton RK, Swaminathan H. Item response theory: Principles and applications. Springer Science & Business Media; 2013 Nov 11.
3. Devlin J, Chang MW, Lee K, Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL-HLT; 2019. p. 4171-4186.
4. Breiman L. Random forests. Machine learning. 2001 Oct;45(1):5-32.
5. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. Advances in neural information processing systems. 2017;30.
6. Domingos P, Pazzani M. On the optimality of the simple Bayesian classifier under zero-one loss. Machine learning. 1997 Nov;29(2):103-30.
7. Richardson L. Beautiful Soup Documentation — Beautiful Soup 4.4.0 documentation. Crummy.com. 2019. Available from: <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>.
8. Scrapy 2.16 documentation — Scrapy 2.16.0 documentation. Scrapy.org. 2026. Available from: <https://docs.scrapy.org/en/latest/>.
9. Van der Linden WJ, Glas CA, editors. Computerized adaptive testing: Theory and practice. Dordrecht: Kluwer Academic; 2000 Jul 31.
10. Thissen D, Steinberg L. A taxonomy of item response models. Psychometrika. 1986 Dec;51(4):567-77.
11. Bellamy RK, Dey K, Hind M, Hoffman SC, Houde S, Kannan K, Lohia P, Martino J, Mehta S, Mojsilovic A, Nagar S. AI Fairness 360: An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias. arXiv preprint arXiv:1810.01943. 2018 Oct 3.
12. Grinberg M. Flask web development. “O'Reilly Media, Inc.”; 2018 Mar 5.
13. Hostinger. Next.js Hosting | Deploy Fast with Hostinger. Hostinger. 2026. Available from: 1 https://www.hostinger.com/in/web-apps-hosting/nextjs-hosting?utm_source=google&utm_medium=cpc&utm_id=23479166145&utm_campaign=Generic-Hosting-Node.js|NT:SE|Lang:EN|LO:IN&utm_term=next.js&utm_content=806857457004&gad_source=1&gad_campaignid=23479166145&gbraid=0AAAAADMy-hY_Vi5n_krleaInMMAc1sK12&gclid=Cj0KCQjwjIPSBhCCARIsABGyK7v6-C7L2f1GffvnlYCEyOXdBjoPdOI-VPkaABstJC7teP_MxXo7koEaAuLrEALw_wcB.
14. Scikit-learn: machine learning in Python — scikit-learn 1.9.0 documentation. Scikit-learn.org. 2026. Available from: <https://scikit-learn.org/stable/>.
15. Supabase. Supabase Docs. Supabase.com. Supabase Docs; 2026. Available from: <https://supabase.com/docs>.