NextStep: Tailored Learning Solutions for Computer Engineering Students

Kalpesh Kolhe[1], Ishaan Naghate[2], Chaitanya Potbhare[3]. Shivam Sonone[4]. Aakash Telharkar[5], Prof. R. D. Thakare[6]

BE Scholars, Department of Information Technology, Sipna College of Engineering & Technology, Amravati, Maharashtra, India

Assistant Professor, Department of Information Technology, Sipna College of Engineering and Technology, Amravati, Maharashtra, India

***Abstract*—In the ever-evolving field of Computer Engineering, students often face the challenge of identifying their strengths and weaknesses when transitioning into technical domains. "NextStep" is a web-based personalized learning and assessment platform designed specifically to address this issue. By offering a comprehensive evaluation system—spanning technical aptitude, theoretical understanding, and non-technical attributes such as personality traits—the system provides a holistic view of a learner's readiness and development areas. Leveraging modern web technologies, adaptive testing frameworks, and psychological assessments like MBTI, the platform generates customized learning directions and data-driven insights, allowing students to take actionable steps in their academic and career journey.**

1. INTRODUCTION

The journey into Computer Engineering demands both technical rigor and self-awareness. While traditional learning systems focus heavily on content delivery, very few platforms personalize the experience based on a student's existing knowledge or cognitive traits. "NextStep" bridges this gap by providing a tailored environment where learners can assess themselves across multiple dimensions—technical knowledge, coding proficiency, and soft traits like cognitive behavior, communication style, and personality alignment with technical roles.

2. OBJECTIVES

The key objectives of the "NextStep" platform include:Helping students identify knowledge gaps across core Computer Engineering subjects, evaluating a user's programming abilities through structured, graded coding tasks, measuring personality-related factors using a customized MBTI-style test, providing adaptive feedback and suggesting relevant next learning steps.

3. LITERATURE REVIEW

In recent years, the integration of adaptive learning systems, psychometric evaluation, and real-time analytics into educational platforms has gained substantial attention. Various studies and technological innovations have contributed foundational principles that directly influence the design and functionality of the *NextStep* system.

[1] Elo Rating System: Originally devised by Arpad Elo for ranking chess players, the Elo rating system has since been adapted for educational purposes. In a learning environment, the Elo model can be used to dynamically adjust a learner's "proficiency score" based on their performance on assessment items of varying difficulty. This helps create a more nuanced understanding of a student's skill level. In the *NextStep* platform, this rating model is applied to domain-specific multiple-choice questions to determine the user's mastery level over time and adapt the test accordingly.

[2] Computerized Adaptive Testing (CAT): CAT, as discussed in *Elements of Adaptive Testing* by van der Linden and Glas, involves tailoring the difficulty of test items in real-time, based on a test taker's performance on previous items. This method enhances test efficiency by reducing the number of questions required to accurately assess a learner's ability. It also maintains engagement by presenting

appropriately challenging questions. *NextStep* incorporates CAT in its Gap Analysis module to personalize the learning assessment journey for each student, ensuring the difficulty curve is optimal for diagnostic accuracy.

[3] Knowledge Space Theory (KST): Developed by Doignon and Falmagne, KST offers a mathematical framework for modeling the dependencies among different knowledge components. It helps determine which concepts a learner is likely to know based on the mastery of prerequisite concepts. The application of KST in *NextStep* allows for intelligent question sequencing in the Gap Analysis Test, ensuring that higher-level questions are only posed if foundational concepts are adequately understood, thus mimicking natural learning progression.

[4] Myers-Briggs Type Indicator (MBTI): The MBTI framework, grounded in Carl Jung's theory of psychological types, categorizes individuals based on cognitive preferences across four dichotomies: Introversion/Extraversion, Sensing/Intuition, Thinking/Feeling, and Judging/Perceiving. The MBTI has been widely adopted in educational and corporate settings to assess personality types and learning styles. *NextStep* employs a customized MBTI-style questionnaire to evaluate students' personality traits, which are then used to generate a coder archetype. This provides valuable insights for role alignment in the software industry—whether the user might thrive in structured problem-solving roles, creative design, or collaborative development.

[5] FastAPI Framework: FastAPI, a modern Python-based web framework, is known for its speed, asynchronous capabilities, and support for RESTful APIs. It facilitates rapid backend development with features like automatic OpenAPI documentation and dependency injection. In the *NextStep* project, FastAPI powers the backend logic that manages authentication, test logic, and database interactions. It enables the delivery of real-time assessment results and ensures the platform remains scalable and efficient.

[6] React.js Frontend Library: React.js, maintained by Meta (formerly Facebook), is a declarative JavaScript library for building interactive user interfaces. Its component-based architecture and virtual DOM optimization make it ideal for dynamic web applications. In *NextStep*, React.js is used to create the frontend components, including the test interfaces, dashboard, and results summary pages. The use of modular design patterns allows for high reusability and ease of maintenance.

[7] SQLAlchemy ORM and SQLite: SQLAlchemy provides a powerful toolkit for database management in Python, while SQLite serves as a lightweight yet robust database engine. The combination of these tools in *NextStep* ensures smooth interaction between application logic and data storage, allowing structured logging of test responses, user profiles, and results in JSON format. This enables efficient data retrieval and analytics for feedback generation.

[8] JWT Authentication Mechanism: JSON Web Tokens (JWT) are widely used for stateless, secure user authentication in web applications. The *NextStep* platform uses JWT for managing user sessions, ensuring that sensitive data—such as test results and personality insights—are accessible only to authenticated users.

[9] Axios HTTP Client: Axios is a promise-based HTTP client that enables seamless communication between the frontend and backend of web applications. It supports features like interceptors and error handling, making it suitable for secure and efficient data exchange. Axios is used in *NextStep* to send test responses and fetch personalized feedback with minimal latency.

[10] GitHub Codespaces: GitHub Codespaces offers cloud-based development environments directly integrated with GitHub repositories. This was the primary development environment for *NextStep*, allowing the team to maintain version control, collaborate remotely, and test the full-stack application in an isolated, production-like setup.

4. TECHNOLOGY STACK

Frontend**:** React.js for developing a dynamic, component-based user interface, React Router for managing navigation and routing within the application, Axios for efficient API calls to the backend, CSS (modular) for styling and interface presentation.

Backend: FastAPI, a modern Python web framework known for high performance, SQLite as the lightweight relational database, interfaced through

SQLAlchemy ORM, JWT (JSON Web Tokens) for secure user authentication and session handling, Modular Python scripts like gap_test.py, init_tech_questions.py, and non_tech_test.py for backend logic.
Deployment Environment: Developed and tested within GitHub Codespaces and Backend hosted on localhost:8000, frontend served via localhost:3000.

5. SYSTEM ARCHITECTURE

The application is structured into two distinct directories: nextstep/frontend: React application including pages and components such as Login, Signup, Dashboard, and each test type, nextstep/backend: FastAPI routes handling user authentication, test logic, and data storage.
This modular architecture promotes maintainability, scalability, and separation of concerns.

6. USER JOURNEY & FUNCTIONAL FLOW

Landing Page: Users are greeted with a minimal interface containing a “Get Started” button, leading to authentication options.
Authentication (Signup/Login): User credentials (name, email, username, password) are collected and stored securely. Authentication is managed via JWTs.
Dashboard: Post-login, users land on a dashboard where they can choose from three test categories: Gap Analysis, Technical Test, and Non-Technical Test.
Gap Analysis Test: Features 60 dynamically selected MCQs from a 160-question pool across 8 Computer Engineering domains, integrates adaptive testing models such as Computerized Adaptive Testing (CAT), Knowledge Space Theory (KST), and Elo rating, Questions categorized as Easy (800), Medium (1200), or Hard (1600) based on Elo, 10-second timer per question enforces recall accuracy, Individual topic Elo scores are tracked to identify weak domains.
Technical Coding Test: Consists of 15 coding questions with incremental complexity, begins with basic tasks (e.g., Hello World) and advances to logic-based problems, Code submissions are analyzed for correctness and minimum structural guidelines, Results tracked as a pair of numbers (e.g., "11,10") indicating solved vs milestone benchmarks.
Non-Technical Test (MBTI-style): Contains 20 Likert-scale questions tailored to the coder's mindset, evaluates four dichotomies (e.g., Introvert vs. Extrovert), Custom-built questions tailored for software development contexts, Plans for adding confidence scoring to enhance MBTI accuracy, Results mapped to a coder personality archetype (e.g., "The Analytical Coder").
Final Report Summary Consolidates test outcomes: Gap Analysis: average Elo scores across topics, Technical Test: solved vs milestone metrics, Non-Technical Test: personality type JSON, Stored within the final_result column in the database.

7. DATABASE DESIGN (ORM - SQLALCHEMY)

User Table Schema: fullname (Text), username (Primary Key), email (Unique), password (Hashed), gap_analysis (JSON), technical_test (Text), non_technical_test (JSON), final_result (JSON)
Question Tables: Technical questions initialized via init_tech_questions.py, Gap Analysis questions categorized by topics and loaded from 8 structured JSON files.

8. CORE ALGORITHMS & FEATURES

Elo Rating Algorithm: Each topic begins with a default Elo of 1200, after every response, the user’s Elo is adjusted based on question difficulty and correctness.
Knowledge Space Theory (KST): Models topic dependencies and prerequisite relationships, Triggers prerequisite-level questions if higher-level performance is low.
Computerized Adaptive Testing (CAT): Dynamically adjusts question difficulty, ensures users are consistently challenged at their competence level.
10-Second Timer Logic: Encourages real-time recall, prevents reliance on external help or excessive time to guess answers.

9. FUTURE SCOPE AND ENHANCEMENTS

Integration of an admin dashboard for managing the question pool, Automated PDF report generation for offline access, Tailored learning recommendations based on detected knowledge gaps, adding user response confidence weighting for improved psychological accuracy in the non-technical test, Deployment on cloud (e.g., Heroku, Render, or AWS EC2).

10. CONCLUSION

"NextStep" serves as an intelligent companion for students aspiring to build a career in Computer Engineering. By offering insights across multiple dimensions—conceptual knowledge, coding skills, and personality alignment—it empowers users to make informed decisions and prioritize areas of development. The use of adaptive testing, scalable web technologies, and psychometric tools make it a versatile and forward-thinking educational aid.

REFERENCES

[1] Elo Rating System Elo, A. E. (1978). The Rating of Chess Players, Past and Present. Arco Publishing.The Elo rating system, originally developed for chess, is widely adopted in adaptive testing to estimate a learner’s proficiency level based on response accuracy and question difficulty.

[2] Computerized Adaptive Testing (CAT)van der Linden, W. J., & Glas, C. A. (Eds.). (2010). Elements of Adaptive Testing. Springer.This work explains the core principles of CAT, where question difficulty dynamically adapts to the learner’s ability, ensuring a personalized assessment experience.

[3] Knowledge Space Theory (KST) Doignon, J. P., & Falmagne, J. C. (1999). Knowledge Spaces. Springer.A foundational theory for modeling and tracking learner knowledge, especially useful in prerequisite-based question selection.

[4] MBTI Personality TypingBriggs Myers, I., McCaulley, M. H., Quenk, N. L., & Hammer, A. L. (1998). MBTI Manual: A Guide to the Development and Use of the Myers-Briggs Type Indicator. Consulting Psychologists Press.The MBTI framework helps assess personality types in terms of cognitive preferences, which has been adapted here for evaluating student work styles in a tech environment.

[5] FastAPI DocumentationSebastián Ramírez. (n.d.). FastAPI: Modern, fast (High-performance), Web Framework for Building APIs with Python 3.6+. Retrieved from https://fastapi.tiangolo.com Official documentation used for building the backend API services efficiently with support for JWT authentication and modular design.

[6] React DocumentationMeta. (n.d.). React – A JavaScript Library for Building User Interfaces. Retrieved from https://reactjs.org Core reference for building component-based frontend architecture for the application.

[7] SQLite and SQLAlchemy ORM SQLite: Hipp, D. R. (2023). SQLite Database Engine. Retrieved from https://sqlite.org. SQLAlchemy: Bayer, M. (n.d.). SQLAlchemy ORM. Retrieved from https://docs.sqlalchemy.orgThese tools were referenced for implementing and managing the application's lightweight database structure and Python ORM integration.

[8] JWT AuthenticationJones, M., Bradley, J., & Sakimura, N. (2015). JSON Web Token (JWT). IETF. Retrieved from https://tools.ietf.org/html/rfc7519JWT was used for secure, stateless authentication and session management across the frontend and backend.

[9] Axios HTTP Client Axios Contributors. (n.d.). Axios – Promise-based HTTP client for the browser and node.js. Retrieved from https://axios-http.com Axios enabled seamless HTTP communication between the React frontend and FastAPI backend.

[10] GitHub Codespaces GitHub. (n.d.). Codespaces – Development Environments in the Cloud. Retrieved from https://github.com/features/codespaces The entire project was developed and tested within GitHub Codespaces, a cloud-based development environment.