SkillsBooster

Enhance your skills with SkillsBooster

**Authors:** Barbara Ferlinc, Klara Kirbiš, Jaša Jernej Rakun Kokalj

**Year:** 2024

# Table of Contents

[Table of Contents 2](#_Toc174920720)

[1 Description 3](#_Toc174920721)

[2 Documentation 3](#_Toc174920722)

[Functionalities 3](#_Toc174920723)

[Architecture 4](#_Toc174920724)

[1. User Interface and Frontend 4](#_Toc174920725)

[2. Backend Components 4](#_Toc174920726)

[3. AI Integration 4](#_Toc174920727)

[4. Deployment 4](#_Toc174920728)

[Teamwork 5](#_Toc174920729)

[1. Methodology 5](#_Toc174920730)

[2. Communication 5](#_Toc174920731)

[3. Tracking process 6](#_Toc174920732)

[4. Division of work 6](#_Toc174920733)

[3 Technologies 6](#_Toc174920734)

[Openai 6](#_Toc174920735)

[Python 6](#_Toc174920736)

[Firebase 6](#_Toc174920737)

[Express Js 7](#_Toc174920738)

[Vite 7](#_Toc174920739)

[React Jsx 7](#_Toc174920740)

[4 Installation and Development 7](#_Toc174920741)

[Installation preview: 7](#_Toc174920742)

[AI 8](#_Toc174920743)

[1. Navigate to the AI Directory 8](#_Toc174920744)

[2. Install Dependencies 8](#_Toc174920745)

[3. Configure Environment Variables 8](#_Toc174920746)

[4. Start the AI Server 8](#_Toc174920747)

[Backend 8](#_Toc174920748)

[Frontend 11](#_Toc174920749)

[5 Testing 12](#_Toc174920750)

[Unit Testing 12](#_Toc174920751)

[Test Scenarios 12](#_Toc174920752)

[Optimization 12](#_Toc174920753)

[6 User Manual 13](#_Toc174920754)

# Description

Our platform offers a comprehensive tool designed to support companies in evaluating their employees' and developing their knowledge and skill sets through customizable self-assessment questionnaires and AI-generated learning content.

The solution aids management in ensuring the continuous growth of their team's skills, knowledge and abilities while providing management with an easy “upload and forget” approach to managing learning resources. Once the content is uploaded, our platform takes care of the rest—automatically integrating it into learning paths for employees. The only thing left for them to do after they have uploaded the learning resources is to click a button and our platform takes care of the rest. This hands-off approach allows management to focus on strategic initiatives, knowing that their workforce's development is being handled efficiently and effectively and that they avoid the burden from the continuous questioning from the employees.

# Documentation

## Functionalities

**Knowledge Domain Management:**

* Easily add and manage knowledge domains
* Define and edit key competencies within each domain

**Question Management**

* User-friendly interface for creating, editing and deleting questions within a domain
* Option to include correct and incorrect answers for each question
* Link questions to theoretical knowledge or relevant learning resources

**AI Integration**

* AI-powered generation of answers to users’ questions
* Analysis of user responses to identify knowledge gaps
* Ability to have specialised ai models that understand and know the knowledge domain
* Analysis of quizzes’ answers and correctness

**Learning Material**

* Link or upload learning materials such as articles, videos, e-books, or web resources relevant to knowledge enhancement.
* Organize learning materials by domain and specific competencies.

**Skills Matrix Generation**

* Track employee progress and compare knowledge levels
* Automatically generate and update a Skills Matrix based on employee’s progress

**Responsiveness and Accessibility**

* Intuitive and user-friendly interface for easy navigation and use
* Accessible on various devices (computers, tablets and smartphones) with adaptability to different screen sizes

**Roles and Functionalities**

* **Administrator:** Can register the company, add users, and assign roles.
* **Manager**: Manages knowledge domains, assigns users to domains, uploads learning materials, creates quizzes, updates learning model, and monitors user progress.
* **User:** Can view their profile, access domains they are added to, view and download uploaded learning material, solve quizzes, and use the AI assistant to help with learning.

## Architecture

### 1. User Interface and Frontend

* **Frontend (React.js):** Users interact with the system through a web-based client-side interface, which is accessible via computers, tablets, and smartphones. This frontend communicates with the backend via HTTP requests and responses.

### 2. Backend Components

* **Web API (Express.js):** The backend, which includes a Web API, handles the main logic of the system. It processes user requests, interacts with the database, and integrates AI capabilities to provide intelligent functionalities like answer checking, answering questions, and suggestions for learning materials.
* **Authentication (Firebase Authentication):** Firebase is used for authenticating users and storing user data. It ensures secure login and data management for different user roles such as administrators, managers, and regular users.
* **Cloud Storage (Cloud Firestore):** Firebase also manages cloud storage for storing data, such as companies, knowledge domains, quizzes, questions and users. It also stores user-uploaded files, which can include learning materials like articles, videos, e-books, etc. These materials are organized by knowledge domain.

### 3. AI Integration

* **AI Model (Python):** An AI model is developed to analyze the files that managers upload and to answer users’ questions. It also examines quizzes completed by users and grades them. Additionally, the AI tracks user progress and automatically refreshes the knowledge matrix, which can be viewed by the manager role.

### 4. Deployment

* **Vercel:** We utilized Vercel to deploy both our backend and frontend seamlessly from our platform. Vercel's powerful infrastructure allows for automatic scaling, optimized performance, and streamlined continuous deployment. This enables us to efficiently deliver a robust, high-performance application with minimal effort, ensuring a smooth user experience and rapid development cycles.
* **Render:** We deployed our AI on Render, leveraging its fully managed cloud platform for seamless deployment and scaling. Render provides automatic scaling, simplified DevOps, and integrated CI/CD, enabling us to focus on developing and optimizing our AI models while ensuring high availability, security, and performance without the overhead of managing infrastructure.

This architecture supports the SkillsBooster platform's goal of evaluating and developing employees’ skills through a combination of user-friendly interfaces, AI-driven insights, and comprehensive management tools.

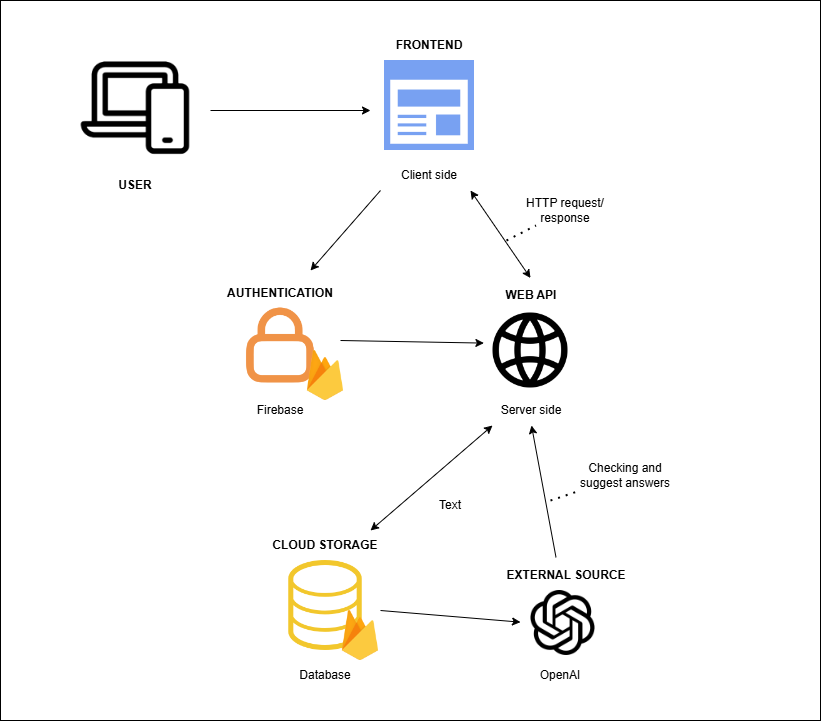


Figure 1 - Architecture diagram from SkillsBooster

## Teamwork

### 1. Methodology

Our team adopted the **Scrum** methodology to enhance our project management and streamline our workflow. Every two weeks, we held sprint planning meetings to define the tasks for the upcoming sprint and assign responsibilities to each team member. This approach ensured that everyone was clear about their roles and the project's priorities, leading to a more organized and focused work environment.

### 2. Communication

Communication was a key component of our Scrum process, and we used **Discord** for all our discussions and weekly check-ins. Discord facilitated real-time conversations and allowed us to address any issues or adjustments promptly. We also scheduled regular meetings to review our progress, discuss roadblocks, and make necessary changes to our plan. This method kept the team aligned and ensured that everyone remained on the same page throughout the sprint.

### 3. Tracking process

To track our tasks and progress, we utilized **Trello**. We categorized our tasks into several boards: "To Do" for upcoming tasks, "Priority" for high-priority items that required immediate attention, "Doing" for tasks currently in progress, and "Done" for completed tasks. This visual system allowed us to manage and prioritize our workload effectively, providing a clear overview of our progress and ensuring that we stayed on track to meet our project goals.

### 4. Division of work

We divided the work among our team members to leverage our individual strengths effectively. One team member focused primarily on the frontend development, ensuring a seamless and user-friendly interface. Meanwhile, two team members concentrated on the backend, with one specializing in AI development and the other handling the remaining backend tasks. Despite our distinct roles, all three of us were actively involved in planning, testing, and optimizing the overall solution. This collaborative approach ensured that we maintained high standards across all aspects of the project, from the user interface to the backend functionality and AI integration.

# Technologies

## Openai

We’ve been dedicated to leveraging OpenAI's technology alongside training our own AI models. OpenAI's advanced models have played a crucial role in enhancing our system, allowing us to analyze files uploaded by managers, learn from their contents, and provide intelligent assistance to users.

Our AI models are categorized into specialized knowledge domains. For instance, in a domain like "statistics," we have a dedicated AI model trained specifically in that area. The study materials provided for students are also used to refine and deepen the AI's understanding of the subject, ensuring continuous improvement. Users can ask questions to the specialized AI, which will deliver the most accurate and insightful answers possible.

The fine-tuning process is automated, requiring only that teachers click the green “Update Model” button to refresh the AI. While the model is being updated, previous versions remain accessible. Additionally, the AI is available to all students through a chatbox at the bottom right corner of the domain, providing instant support for their questions. It is also capable of examining and grading quizzes completed by users.

## Python

In our project, we utilized Python to write the code for training our AI models. Python’s robust ecosystem of libraries and frameworks made it an ideal choice for developing and fine-tuning machine learning algorithms. Python’s versatility and extensive support for machine learning facilitated an efficient and effective model training process.

## Firebase

We extensively utilized Firebase to support various functionalities within our platform. For user management, we implemented Firebase Authentication, which allowed users to register, log in, and log out using their email and password. This service also enabled us to manage session durations effectively, ensuring users were automatically logged out after a specified period of inactivity, enhancing both security and user experience.

Additionally, Firebase Storage was used to manage and store educational materials in dynamically added knowledge domains. This service enabled us to upload and securely store files contributed by company managers, ensuring that essential resources were readily accessible and well-organized.

We utilized Firebase Cloud Firestore for data storage, where we organized and maintained our entities. This NoSQL database allowed us to handle and query our data in a flexible and scalable manner, supporting the dynamic needs of our application. Our database structure included various collections and documents, each with specific attributes tailored to our application’s requirements:

* Companies: address, admin, name, postal\_code
* Knowledge\_domains: description, employees, key\_skills, learning\_materials, links, name, owner, quizzes, results
* Questions: answers, question, quiz, type
* Quizzes: name, questions, results
* Users: admin, email, full\_name, password (hashed), role

## Express Js

For the backend, we utilized Express.js, a minimal and flexible Node.js web application framework. Express.js provided a robust set of features for building server-side applications and APIs, streamlining our development process. Its middleware architecture allowed us to handle HTTP requests, manage routes, and integrate various functionalities seamlessly, facilitating a smooth and efficient backend development experience.

## Vite

We used Vite as our modern build tool for the frontend, enhancing the development experience for our web applications. Developed by the creator of Vue.js, Vite addresses the performance limitations of traditional build tools like Webpack, particularly in large projects. Its fast build times and efficient development workflow allowed us to streamline our frontend development process, making it more responsive and productive.

## React Jsx

We employed React JSX for building our user interfaces. JSX is a syntax extension for JavaScript that enables us to write HTML-like code directly within JavaScript files. This approach simplifies the creation and manipulation of UI components in a declarative manner, allowing us to build complex interfaces more intuitively and efficiently.

# Installation and Development

## Installation preview:

Installed **Node.js** and **cloned GitHub repository**

**1.Clone repository**

First, you need to clone the project's repository from the version control system (e.g., GitHub).

Use the following command in your terminal:

git clone <repository-url>

Replace *<repository-url>* with the actual URL of the repository. This command will download a copy of the project's files into a folder on your local machine.

**1.2 Navigate to the Project Directory**

1. Once the repository is cloned, move into the project’s directory:

cd <project-directory>

2. Replace <project-directory> with the name of the folder that was created when you cloned the repository.

## AI

To get started with the AI component of the project, follow these steps:

### 1. Navigate to the AI Directory

First, access the ‘backend/ai’ directory of your project.

### 2. Install Dependencies

Install all the necessary Python packages by running the following command:

pip install –r requirements.txt

This will ensure that all required libraries are installed.

### 3. Configure Environment Variables

Create an **.env** file in the **backend/ai** directory. This file should contain your OpenAI API key, formatted as follows:

# AI Configuration

OPENAI\_API\_KEY= <your-openai-api-key>

### 4. Start the AI Server

To run the AI server, use the command:

uvicorn app:app --host 0.0.0.0 --port 8000 --timeout-keep-alive 3000

This command will start the server on port 8000, allowing it to handle requests as configured.

## Backend

**1. Navigate to backend directory**:

cd backend

**2. Install necessary dependencies**

1. Install all the required node\_modules (JavaScript libraries and dependencies) for the backend by running:

npm install

This command reads the package.json file and installs all the dependencies listed under the dependencies section.

**3. Configure Environment Variables**

To properly configure the backend, you need to upload a key.json file from your Firebase project into the backend directory. This file contains the credentials necessary for Firebase services.

Additionally, **create an .env file** in the **backend directory**. This file will hold all the necessary environment variables that the backend requires to function, such as API keys and configuration settings.

The **.env** file should include the following variables:

# Access CORS

ACCESS\_CORS= <frontend-url>

# Firebase configuration

EXPRESS\_APP\_API\_KEY= <firebase-api-key>

EXPRESS\_APP\_AUTH\_DOMAIN= <firebase-auth-domain>

EXPRESS\_APP\_PROJECT\_ID= <firebase-project-id>

EXPRESS\_APP\_STORAGE\_BUCKET= <firebase-storage-bucket>

EXPRESS\_APP\_MESSAGING\_SENDER\_ID= <firebase-messaging-sender-id>

EXPRESS\_APP\_APP\_ID= <firebase-app-id>

# AI configuration

OPENAI\_FINETUNE\_URL= <openai-finetunig-url>

OPENAI\_API\_KEY= <openai-api-key>

OPENAI\_BACKUP\_MODEL= <openai-basic-model>

# SMTP configuration

SMTP\_HOST= <smtp-host>

SMTP\_PORT= <smtp-port>

SMTP\_USER= <smtp-email>

SMTP\_PASSWORD= <smtp-password>

Replace each placeholder (<frontend-url>, <firebase-api-key>, etc.) with the actual values specific to your Firebase and OpenAI projects.

**4. Start Backend Server**

To start the backend server, run:

node server.js

This will start the server on port 9000. You should now have a running backend server that your frontend application can interact with.

5. Testing the backend

If you want to run unit tests to ensure that the backend logic is functioning correctly, use the following command:

npm test

This will execute all tests defined in your backend project, allowing you to verify that everything works as expected.

## Frontend

**1. Navigate to the frontend directory**

Go back to the main project directory, then move to the frontend directory:

cd ../frontend

**2. Install necessary dependencies**

Install all the required node\_modules for the frontend by running:

npm install

Like the backend, this command will install all the frontend dependencies specified in the package.json file.

**3. Configure Environment variables:**

In the frontend directory, **create an .env** file with the necessary configuration settings for your Firebase project.

The **.env** file should include the following variables:

# Link to backend

VITE\_APP\_BASE\_URL= <backend-url>

# Firebase configuration

VITE\_API\_KEY= <firebase-api-key>

VITE\_AUTH\_DOMAIN= <firebase-auth-domain>

VITE\_PROJECT\_ID= <firebase-project-id>

VITE\_STORAGE\_BUCKET= <firebase-storage-bucket>

VITE\_MESSAGING\_SENDER\_ID= <firebase-messaging-sender-id>

VITE\_APP\_ID= <firebase-app-id>

Replace each placeholder (<backend-url>, <firebase-api-key>, etc.) with the appropriate values.

**4. Start frontend development server**

To start the development server for the frontend, run:

npm run dev

**5. Building for production**

When you're ready to deploy the frontend, create a production build by running:

npm run build

This command optimizes and bundles the application files, making them ready for deployment on a web server.

# Testing

## Unit Testing

In our project, we implemented unit tests to ensure the reliability and correctness of our code. These tests are organized in the backend/tests directory and cover a comprehensive range of functionality. Each test is designed to validate the behavior of individual entities and core functionalities, providing a robust framework for detecting and addressing issues early in the development process. By systematically testing each component, we ensure that our system operates as expected and maintains high standards of quality throughout the project's lifecycle.

## Test Scenarios

## Optimization

To optimize our solution, we integrated SonarCloud with our project through GitHub. SonarCloud provided valuable insights into code quality and technical debt, enabling us to identify and address potential issues early in the development cycle. By linking our GitHub repository with SonarCloud, we ensured continuous analysis and feedback, which helped us maintain high standards of code quality and improve overall project efficiency.

With SonarCloud integrated into our GitHub project, we systematically monitored various aspects of code quality and health. SonarCloud provided in-depth analysis of code coverage, identifying areas where tests were lacking or incomplete. It also flagged code smells, highlighting areas where the code could be refactored for better readability and maintainability. Additionally, it assessed code duplication, helping us eliminate redundant code and improve efficiency. Security vulnerabilities were detected to ensure our codebase was safeguarded against potential threats.

While SonarCloud provided valuable insights, we accepted certain issues that did not impact the overall functionality or performance of the code. These exceptions were carefully evaluated to ensure they did not compromise the project's integrity. By continuously reviewing these metrics and making informed decisions about what to address, SonarCloud played a crucial role in maintaining high standards for our codebase, supporting a robust, secure, and well-maintained project.

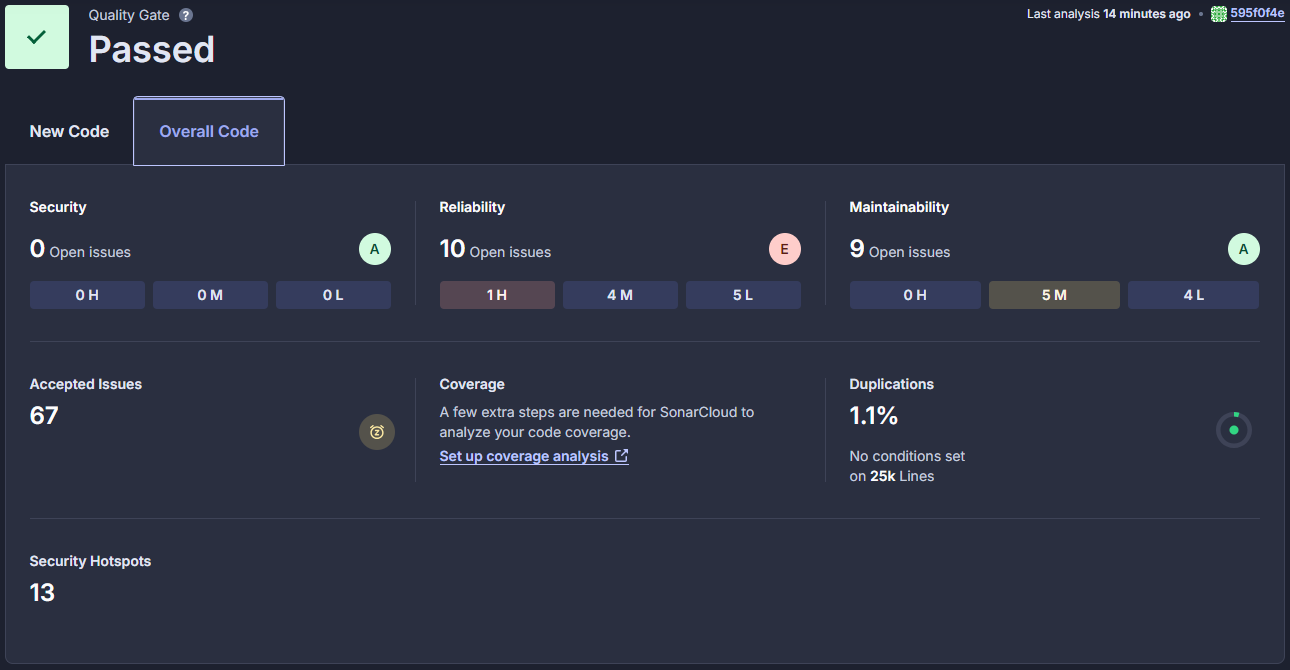


Figure 2 - Analysis results from SonarCloud for SkillsBooster

# User Manual

You can find User Manual in a separate file “SkillsBooster User Manual” in the folder: Documentation, on our repository, where everything is described in detail on how to use our website.