



Design | Technology | Solutions | Helloindiasolutions.com

TECHNICAL SERVICE
ARCHITECTURE
DOCUMENTATION

June 23 , 2024

This document provides a detailed description of the back-end architecture for the Students Learning Platform. This platform is a critical application designed to enhance the learning experience by integrating and managing core educational processes..

Scope

This document covers the Frontend and Backend Over through of the Students Learning Platform Reviewed by Helloindiasolutions . It includes:

- High-level and detailed architecture diagrams
- Descriptions of all back-end components and modules
- Data storage and management strategies
- Security protocols and measures
- Integration points with other internal and external systems
- Scalability, performance, and reliability considerations

Objectives

- **Enhance Student Learning:** Provide a comprehensive and engaging platform that supports students in their educational journey by offering diverse resources and tools.
- **Seamless Integration:** Ensure smooth integration with various internal and external systems to provide a unified learning experience.
- **Scalability and Performance:** Design a system that can handle increasing numbers of users and data without compromising on performance.

PART 1 Frontend Standard

○ Introduction	1
○ Technology Stack	2
▪ Overview	
▪ Development Standards and Practices	
▪ Dependencies Modules	3
○ Components Stack	4
▪ Overview	6
▪ Component Module Strength	7
○ Directories and Files	8
○ Project Code Structure	9
○ Styles & Assets	13
○ Internationalization	15

PART 2 Backend Standard

○ Enhance Hackathon Platform System	17
○ Component Details	18
○ Data Flow Diagram	19
○ Database Design	20
○ Security Architecture	29
○ Environment Architecture	30
○ Dependencies	31
○ Configuration Management	33

The frontend of the TNEDII Hackathon Platform is designed to offer a seamless and feature-rich interface for users interacting with educational content and management tools. It leverages components, state management, and API integration to ensure efficient data retrieval and manipulation. Whether you're a student accessing course materials or an instructor managing curriculum, the frontend provides a unified and intuitive experience.

Frontend Standard

Introduction to Frontend Development Standards

Frontend development in This project is meticulously designed to deliver a seamless and engaging user experience through React, a powerful JavaScript library. This introduction outlines our commitment to best practices, scalability, and user-centric design principles across various modules and components. Frontend development serves as the user-facing interface of This application, directly impacting user interaction, usability, and overall satisfaction. It encompasses a wide array of technologies, libraries, and methodologies to ensure robust performance, maintainability, and extensibility over time.

Coding Conventions

Consistency in coding conventions is paramount for clarity and collaboration. Our team adheres to industry best practices such as meaningful variable naming, consistent formatting, and adherence to ES6+ syntax. This approach not only enhances code readability but also reduces potential bugs and ensures maintainability throughout the project's lifecycle.

UI Guidelines

UI guidelines define the visual and interactive components across this applications. We prioritize responsiveness, accessibility, and intuitive design principles to create an engaging user experience. Components are designed with a focus on user interface consistency, ensuring seamless navigation and clear information presentation.

Technology Stack

Overview

This frontend technology stack revolves around React, chosen for its component-based architecture, declarative syntax, and efficient rendering capabilities. Leveraging React allows us to create reusable UI components, resulting in a modular, efficient, and maintainable codebase. At the core of frontend development strategy lies user-centric design principles. We prioritize intuitive navigation, responsive layouts, and accessibility to cater to diverse user needs and preferences. Through thoughtful UI/UX design, we aim to enhance user engagement and satisfaction across the platform .

Performance is a critical aspect of Our frontend Review process. We employ techniques such as code splitting, lazy loading, and caching to minimize load times and ensure smooth user interactions. By optimizing performance, we prioritize efficiency and responsiveness, even under heavy traffic conditions. Scalability is key to accommodating future growth and feature expansions. This frontend architecture is designed to scale seamlessly, supporting additional functionalities and accommodating increased user traffic without compromising performance or user experience.

Development Standards and Practices

Consistency and adherence to coding standards are paramount. We follow industry best practices such as modular CSS, component reusability, and version control to maintain code quality and facilitate collaborative development. Through continuous integration and automated testing, we ensure code reliability and minimize errors before deployment. This frontend supports internationalization (i18n) and localization (l10n), enabling users worldwide to access content in their preferred languages. This inclusivity fosters a global user base and enhances accessibility, reflecting our commitment to diversity and user inclusiveness.

To enrich functionality and streamline development, we integrate various third-party libraries and APIs. These include charting libraries for data visualization, form validation tools, and UI component libraries like Ant Design and Bootstrap. By leveraging these resources, we enhance productivity and accelerate feature implementation while maintaining high standards of quality.

Dependencies Modules

React	: Provides the foundational library for building user interfaces in This project.
Ant Design Icons	: Offers a comprehensive suite of icons to enhance the visual appeal and functionality of components.
Axios	: Facilitates HTTP requests for data fetching and communication with backend services.
Bootstrap	: Provides UI components and utilities for responsive and mobile-first design.
Chart.js	: Enables data visualization through interactive charts and graphs.
Yup	: A schema validation library for form validation and data integrity.
Redux	:Manages application state and facilitates predictable state management.
React Router DOM	: Enables navigation and routing within the React application.
Styled Components	: Allows styling components with scoped CSS and enhances component
Sass	: Provides CSS extension features such as variables, mixins, and nested rules for enhanced styling capabilities.
I18next	: Facilitates internationalization (i18n) and localization (l10n) for multilingual support across the application.
React Datepicker	: Offers a customizable date picker component for date selection and input

Component Stack

Overview

The project is organized into several key sections, each containing components specific to different user roles and functionalities. The components are grouped based on their purpose and user role, such as Student, Admin, Teacher, Evaluator, etc. Each component is responsible for a specific part of the application's UI and behavior.

1. Admin

The Admin component serves as the backbone of administrative tasks within This platform. It provides privileged access to manage user accounts, permissions, and system configurations. Admins can efficiently monitor and control various aspects of the application through a user-friendly interface, ensuring smooth operations and data management.

2. Badges

Badges are visual indicators of achievements and milestones attained by users. In This project, badges serve to gamify user engagement, providing incentives and recognition for completing tasks, reaching goals, or mastering specific skills. Each badge is meticulously designed to motivate users and enhance their journey within the platform.

3. Challenges

Challenges feature prominently in This platform to foster skill development and creativity among users. Users can participate in various challenges tailored to different skills or themes, showcasing their abilities through submissions and receiving feedback from peers or evaluators. The Challenges component includes features for creating, managing, and participating in challenges, ensuring an engaging and rewarding experience.

4. Courses

Courses represent structured learning paths offered within This platform. Each course is designed to impart knowledge and skills on specific topics or subjects, catering to diverse learning needs and preferences. Users can enroll, track progress, and access course materials such as lectures, quizzes, and assignments, supported by interactive features to enhance learning effectiveness.

5. Dashboard

The Dashboard serves as the central hub for users to access personalized information, analytics, and relevant updates. It provides a comprehensive overview of user activity, progress, and notifications, facilitating efficient navigation and interaction within the platform.

6. Detailed Quiz

Detailed Quiz functionality offers comprehensive assessments tailored to specific topics or subjects. Users can engage in quizzes with detailed feedback, performance analytics, and adaptive questioning based on their responses. The Detailed Quiz component ensures an interactive and educational experience, enabling users to gauge their knowledge and track improvement over time.

7. Evaluation Process

The Evaluation Process component manages the assessment and review workflows within This platform. It facilitates the evaluation of submissions, assignments, or projects based on predefined criteria and standards. Evaluators can provide structured feedback, assign ratings, and communicate outcomes effectively, ensuring transparency and fairness in the evaluation process.

8. Evaluation

Evaluation functionality encompasses the overall assessment of user submissions or achievements. It integrates evaluation criteria, scoring mechanisms, and feedback loops to measure performance and proficiency accurately. The Evaluation component supports continuous improvement by identifying strengths, areas for improvement, and providing actionable insights for users and administrators.

9. FAQ

The FAQ (Frequently Asked Questions) component offers a repository of common queries and answers to aid users in navigating the platform. It serves as a knowledge base, providing clarity on features, policies, and troubleshooting steps. Users can quickly find solutions to their queries, reducing support requests and enhancing user satisfaction.

10. Latest News

Latest News keeps users informed about platform updates, announcements, and relevant industry news. It delivers timely information through curated articles, blog posts, or notifications, ensuring users stay abreast of developments that impact their experience. The Latest News component enhances user engagement and fosters a sense of community within the platform.

11. Popup

Popups are used strategically to deliver important messages, prompts, or notifications to users. They enhance user interaction by displaying relevant content or actions without disrupting workflow. The Popup component ensures effective communication, engagement, and user responsiveness, contributing to a seamless user experience.

12. Quiz

The Quiz component offers interactive assessments and quizzes designed to test users' knowledge and skills. It supports various question formats, timers, and scoring mechanisms to engage users in challenging yet rewarding learning experiences. Real-time feedback and performance tracking enhance learning outcomes and encourage continuous learning.

13. Report

Reporting functionality enables users to generate and view detailed reports based on performance metrics, analytics, or specific criteria. Reports can include graphical representations, data summaries, and insights to facilitate decision-making and progress monitoring. The Report component empowers users to extract valuable information and derive actionable insights from platform data.

14. Resources

Resources provide users with access to educational materials, reference documents, or multimedia content relevant to their learning or professional development. The Resources component organizes content into categories, facilitates search and retrieval, and ensures easy navigation for users seeking valuable information and resources.

15. Schools

The Schools component focuses on managing educational institutions within the platform. It includes features for school registration, profile management, and administrative functions tailored to school administrators and staff. Users can access school-specific information, updates, and resources, fostering collaboration and support among educational stakeholders.

16. Store

The Store component facilitates transactions and interactions related to products, services, or digital goods offered within the platform. It includes features for browsing, purchasing, and managing items, ensuring a seamless shopping experience for users. The Store component supports e-commerce functionalities, order management, and secure payment processing.

17. Test

The Test component offers assessments, examinations, or proficiency tests designed to measure users' knowledge and skills in specific areas. It supports various test formats, adaptive questioning, and automated grading to provide accurate assessments and feedback. The Test component enhances learning outcomes and supports certification or credentialing processes.

18. Tickets

Tickets manage user inquiries, issues, or support requests within the platform. They streamline communication between users and support teams, ensuring prompt resolution of queries or technical issues. The Tickets component includes features for ticket creation, status tracking, and escalation, enhancing customer support and user satisfaction.

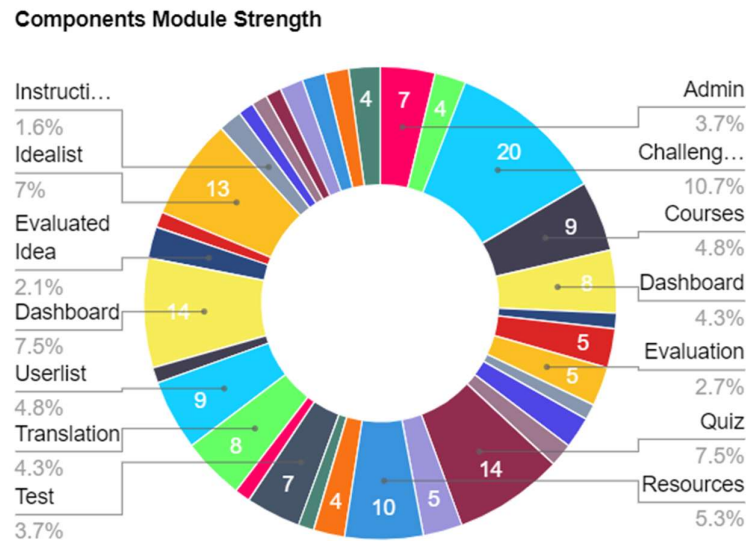
19. Translation

Translation functionality enables multi-language support within the platform, catering to users from diverse linguistic backgrounds. It includes features for language selection, content localization, and translation management, ensuring accessibility and inclusivity. The Translation component facilitates global user engagement and enhances usability across international markets.

20. Userlist

The Userlist component provides administrators with a comprehensive view of platform users, their roles, and permissions. It supports user management tasks such as registration, profile updates, and access control, ensuring efficient administration and governance. The Userlist component enhances platform security, user experience, and administrative oversight.

Component Module Strength



Directories and Files

Overview

The project is organized into several main directories, each serving a specific purpose. Below is an overview of the key directories and their contents:

Directories

- **src/**
- **Admin/** - Components and pages related to admin functionalities.
- **Coordinators/** - Components and pages for coordinators.
- **Evaluator/** - Components and pages for evaluators.
- **Helpers/** - Helper functions and utilities.
- **Landing_page/** - Components for the landing page.
- **RegPage/** - Registration related components.
- **ReportsPanel/** - Reporting components and pages.
- **School/** - Components related to school functionalities.
- **Student/** - Components and pages for student functionalities.
- **SupportingSCSS/** - SCSS stylesheets.
- **Teachers/** - Components and pages for teachers.
- **i18n.js** - Internationalization configuration.
- **index.js** - Entry point of the application.
- **App.js** - Main application component.
- **Routes.js** - Routing configuration.

Project Code Structure

Overview

The project is organized into several directories, each serving a specific purpose. This structure helps in maintaining a clean and organized codebase, facilitating easier development and maintenance.

Project Root

```
|— src
|   |— Admin
|   |   |— Challenges
|   |   |   |— ViewSelectedChallenges.js
|   |   |— Dashboard
|   |   |   |— index.js
|   |   |   |— ViewMore.js
|   |   |— Evaluation
|   |   |   |— indexStatic.js
|   |   |   |— index.js
|   |   |   |— ViewSelectedIdea
|   |   |       |— ViewSelectedIdeas.js
|   |   |   |— FinalResults
|   |   |       |— ViewFinalSelectedIdeas.js
|   |   |— FAQ
|   |   |   |— AddNewFaq.js
|   |   |   |— EditFaq.js
|   |   |       |— FaqByCategory.js
|   |   |— LatestNews
|   |   |   |— TeacherNews.js
|   |   |   |— createLatestNews.js
|   |   |       |— editLatestNews.js
|   |   |— LoginNew.js
|   |   |— MyProfile.js
|   |   |— Popup
|   |       |— popup.js
```

- | | └─ Reports
- | | | └─ index.js
- | | | └─ Helpers
- | | | └─ ReportsView.js
- | | | └─ ReportsRegistration.js
- | | | └─ SurveyStatus.js
- | | | └─ TeacherProgressDetailed.js
- | | | └─ StudentsProgressReport.js
- | | | └─ ChallengesReport.js
- | | | └─ IdeasDetailsReport.js
- | | | └─ InstDetailsReport.js
- | | | └─ IdeaSubmittedReport.js
- | | | └─ IdeaEvaluationReport.js
- | | | └─ SubIdeasTableReports.js
- | | | └─ DistAbstractReport.js
- | | | └─ InstTypePerformance.js
- | | | └─ InstWiseReport.js
- | | | └─ YearWiseReport.js
- | | | └─ ReportFilter.js
- | | └─ Resources
- | | | └─ ResourcesList.js
- | | | └─ createResource.js
- | | | └─ editResource.js
- | | └─ Schools
- | | | └─ AddNewSchool.js
- | | | └─ Ticket.js
- | | | └─ EditSchool.js
- | | └─ Translation
- | | | └─ Translation.js
- | | | └─ EditTranslation.js
- | | | └─ CreateTranslation.js
- | | └─ Tickets
- | | | └─ Ticket.js
- | | | └─ TicketResView.js
- | | | └─ TicketResponse.js

- | | | — UserList
- | | | | — Ticket.js
- | | | | — CommonUserProfile.js
- | | | | — EditProfile.js
- | | | | — TeacherViewDetails.js
- | | | | — AdminTeacherDashboard.js
- | | | | — MentorEditProfile.js
- | | | | — StudentEditProfile.js
- | | | | — TeacherRegister.js
- | | | | — SuccessTeacher.js
- | | | | — AtlReg.js
- | | | | — NonAtlReg.js
- | | | | — SuccessNonAtl.js
- | | | | — SuccessAtl.js
- | | | — ForgotPassword.js
- | | — Coordi...
- | | — Evaluator
- | | — Landing_page
- | | — Register
- | | — RegPage
- | | — ReportsPanel
- | | — School
- | | — Student
- | | | — Pages
- | | | — Resources
- | | | — Certificate
- | | | — PreSurvey
- | | — Teachers
- | | | — Certificate
- | | | — Courses
- | | | — Dashboard
- | | | — HelpPages
- | | | — MyProfile
- | | | — PreSurvey
- | | | — PostSurvey

```
| | |— Resources
| | |— SupportJourney
| | |— Teams
| | |— ChangePSWModal.js
| |— helpers
| |— SupportingSCSS
| |— i18n.js
|— public
| |— index.html
|— index.js
```

Styles And Assets

Overview

The styling of the application is managed using SCSS (Sass), which provides a modular and maintainable approach to CSS. The styles are organized in the SupportingSCSS directory to ensure a clear and logical structure.

SupportingSCSS Directory Structure

base/	- Contains base styles such as resets, typography, and global settings.
components/	- Component-specific styles, organized by component name.
layouts/	- Layout-specific styles, such as header, footer, and grid systems.
pages/	- Page-specific styles, organized by page name.
themes/	- Theme styles for different themes if applicable (light, dark, etc.).
utils/	- Utility classes and mixins for reuse across different stylesheets.
variables/	- SCSS variables for colors, fonts, spacing, and other design tokens.

Key Files

_reset.scss	- Resets default browser styling to ensure consistency across different browsers.
_variables.scss	- Contains design tokens like colors, fonts, and spacing, enabling easy theme management.
_mixins.scss	- Reusable mixins for common CSS patterns, promoting DRY (Don't Repeat Yourself) principles.
_global.scss	- Global styles that apply to the entire application, such as body and heading styles.

Assets Directory Structure

- images/** - Contains all image files used in the application, further organized into subdirectories based on usage (e.g., icons/, backgrounds/, logos/).
- fonts/** - Custom fonts used in the application.
- icons/** - SVG or other icon files, often used for UI elements.

Best Practices for Assets

- Optimization** : Images and other assets are optimized for web performance to ensure fast load times.
- Naming Conventions** : Files are named descriptively to indicate their purpose and usage, making it easy to locate specific assets.
- Consistency** : Consistent use of file formats and sizes helps maintain a uniform appearance across the application.

Example SCSS File

Here's an example of a typical SCSS file structure for a component:

```
@import '../variables';
@import '../mixins';

.button {
  background-color: $primary-color;
  color: $text-color;
  padding: $spacing-medium;
  border-radius: $border-radius;
  @include transition(all 0.3s ease);
}
```

This SCSS file for a button component demonstrates the use of variables, mixins, and BEM (Block Element Modifier) naming conventions, promoting consistency and reusability.

Internationalization

Overview

Internationalization (i18n) is the process of designing and preparing your application to support various languages and regional settings without requiring engineering changes. This project uses the i18next library, a popular internationalization framework for React, to handle multiple languages efficiently.

Implementation Setup

The i18n setup is typically configured in a dedicated file, such as i18n.js. This file initializes the i18next library, sets up language resources, and configures language detection and fallback options.

```
// i18n.js

import i18n from 'i18next';
import { initReactI18next } from 'react-i18next';
import translationEN from './locales/en/translation.json';
import translationES from './locales/es/translation.json';

const resources = {
  en: {
    translation: translationEN,
  },
  es: {
    translation: translationES,
  },
};

i18n
  .use(initReactI18next)
  .init({
    resources,
    lng: 'en', // Default language
    fallbackLng: 'en', // Fallback language
```

```
interpolation: {  
  escapeValue: false,  
},  
});
```

```
export default i18n;
```

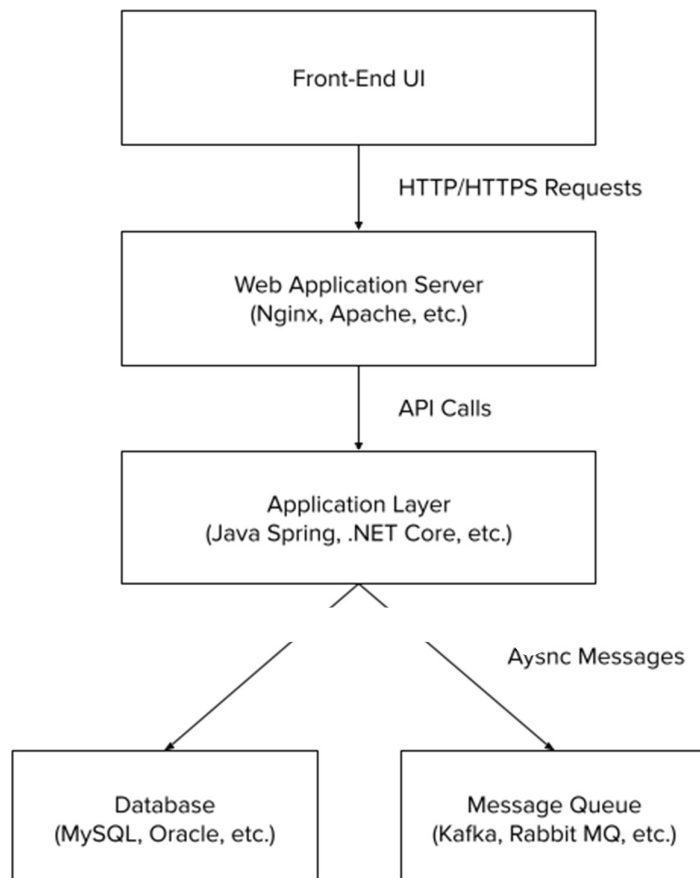
Directory Structure

locales/	- Contains language-specific directories with translation files.
en/	- English translations.
translation.json	- JSON file with key-value pairs for English translations.
es/	- Spanish translations.
translation.json	- JSON file with key-value pairs for Spanish translations.

The backend of the Student's Learning Platform is designed to support a range of features essential for a modern educational platform. It offers a well-structured API to facilitate various functionalities such as user authentication, CRUD operations, data validation, and logging. The primary goals of this backend are to ensure security, efficiency, and maintainability while providing a seamless experience for developers and end-users.

Enhance Hackathon Platform System

The following diagram showcases the major components of the system and their interactions.



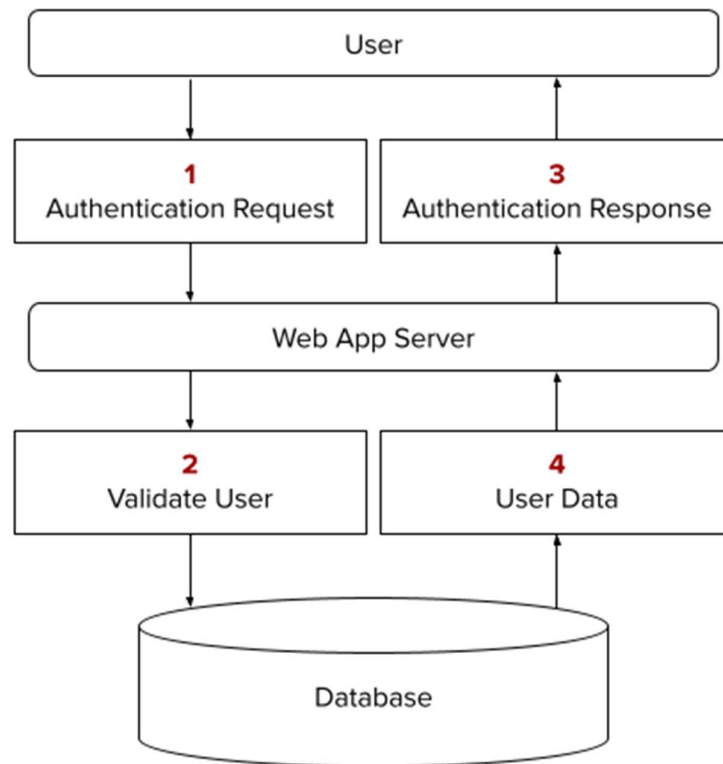
Component Details

Below are the detailed descriptions of each major component/module in Hackathon Platform System, including their purpose, functionality, and interactions with other components:

	Purpose	Functionality	Interaction with Other Components
Front-End UI	To provide end-users with an interactive and user-friendly interface to interact	<ul style="list-style-type: none"> ● User authentication and authorization ● Displaying data and reports 	<ul style="list-style-type: none"> ● Sends HTTP/HTTPS requests to the Web Application Server
Web Application Server	To manage and handle incoming HTTP/HTTPS requests and route them to the appropriate services in the Application Layer.	<ul style="list-style-type: none"> ● SSL termination ● Load balancing ● Request routing ● Static content serving ● Caching 	<ul style="list-style-type: none"> ● Receives requests from the Front-End UI ● Forwards API calls to the Application Layer
Application Layer	To contain the core business logic of the ERP System.	<ul style="list-style-type: none"> ● Performing CRUD (Create, Read, Update, Delete) operations on data 	<ul style="list-style-type: none"> ● Receives API calls from the Web Application Server ● Reads from and writes to the Database
Database	To store, organize, and manage data	<ul style="list-style-type: none"> ● Data storage and retrieval ● Data indexing for efficient querying 	<ul style="list-style-type: none"> ● Application Layer performs CRUD operations on the Database

Data Flow Diagrams

1. User Authentication Flow



1. The user sends an authentication request with credentials to the Web Application Server.
2. The Web Application Server sends a request to validate the user credentials to the Database.
3. After validation, the Web Application Server sends an authentication response back to the user.
4. The Database sends user data back to the Web Application Server.

Database Design

This section provides an overview of the database schema design for the IT Corp's ERP System, including tables, relationships, and indexes used to store and manage the application's data efficiently.

Table	Fields	Indexes	Description
Admins	admin_id (PK) user_id full_name date_of_birth city district state country status created_by created_at updated_by updated_at	Index on User_id	This table stores information about each user who has access to the Admin
Badges	badge_id (PK) slug name desc icon status created_by created_at updated_by updated_at	Index on Slug Name	This table holds all the Badges on this system
Blocks	block_id (Primary Key) district_id (Indexed) block_name block_name_vernacular ldg_block_code status created_by created_at updated_by updated_at	Index on District_id	This table contains information about the Block level District
Certificate download	certificate_download_id (Primary Key) certificate_type mobile (Indexed) faculty_name organization_name status created_by created_at updated_by updated_at	Composite Index on Mobile	This table links orders with the products they contain, effectively representing the items in an order.
Challenges	challenge_id (Primary Key) name status created_by	Index on Name	This table stores information about each Challenges

	created_at updated_by updated_at		
Challenges_evaluator_map	challenge_evaluator_id (Primary Key) evaluator_id (Indexed) challenge_response_id status created_by created_at updated_by updated_at	Index on Evaluator	This table stores information about each Challenges Evaluations
Challenges_Questions	challenge_question_id (Primary Key) challenge_id (Indexed) question_no (Indexed) question description option_a option_b option_c option_d correct_ans created_by created_at updated_by updated_at word_limit	Index on Question_no	This table stores information about each Questions Challenges
Challenge_responses	challenge_response_id (Primary Key) challenge_id (Indexed) team_id (Indexed) response initiated_by status ('DRAFT', 'SUBMITTED', 'EVALUATION', 'SELECTED...') created_by created_at updated_by updated_at sdg sub_category others evaluated_by evaluated_at submitted_at evaluation_status ('SELECTEDROUND1', 'REJECTEDROUND1') rejected_reason rejected_reasonSecond district state final_result ('0', '1') manual_update	Index on Challenge Team_id	This table stores information about each Challenges Responses
Courses	course_id (Primary Key) title description thumbnail (default value: '/images/default.jpg') status ('ACTIVE', 'INACTIVE', 'DELETED', 'LOCKED') created_by created_at updated_by	Index on None	This table stores information about each Courses

	updated_at		
Courses_modules	course_module_id (Primary Key) course_id (Index) title description status (enum: 'ACTIVE', 'INACTIVE', 'DELETED', 'LOCKED') created_by created_at updated_by updated_at	Index on Course Id	This table stores information about each Courses Modules
Course Topics	course_topic_id (Primary Key) course_module_id (Index) topic_type_id (Index) status (enum: 'ACTIVE', 'INACTIVE', 'DELETED', 'LOCKED') created_by created_at updated_by updated_at	Index on Course Module_id Topic_type Id	This table stores information about each Course Topics
Dashboard_map_state	dashboard_map_stat_id (Primary Key) state_name overall_schools reg_schools teams students ideas reg_mentors schools_with_teams status (enum: 'ACTIVE', 'INACTIVE', 'DELETED', 'LOCKED') created_by created_at updated_by updated_at	Index on None	This table stores information about each Dashboard Map State
District	district_id (Primary Key) state_id (Index) lgd_code district_name (Index) district_name_vernacular district_short_form district_headquarters district_headquarters_vernacular status created_by created_at updated_by updated_at	Index on State Id	This table stores information about each District
Evaluation Process	evaluation_process_id (Primary Key) level_name no_of_evaluation eval_schema district status created_by created_at updated_by updated_at	Index on None	This table stores information about each Evaluation Process

Evaluation Result	evaluation_results_id (Primary Key) idea_id (Indexed) level status created_by created_at updated_by updated_at	Index on Idea_id	This table stores information about each Evaluation Result
Evaluators	evaluator_id (Primary Key, Auto Increment) user_id (Indexed) full_name district mobile (Indexed) status (Enum: ACTIVE, INACTIVE, DELETED, LOCKED, ...) created_by created_at updated_by updated_at	Index on User_id	This table stores information about each Evaluators
Evaluators Rating	evaluator_rating_id (Primary Key, Auto Increment) evaluator_id (Indexed) idea_id (Indexed) status (Enum: ACTIVE, INACTIVE, DELETED, LOCKED, ...) level param_1 param_2 param_3 param_4 param_5 comments overall (Decimal with precision 10 and scale 0) submitted_at created_by created_at updated_by updated_at	Index on Evaluator Idea_id	This table stores information about each Evaluators Rating
Faqs	faq_id (Primary Key, Auto Increment) faq_category_id (Indexed) question (Long Text) answer (Long Text) status (Enum: ACTIVE, INACTIVE, DELETED, LOCKED) created_by created_at updated_by updated_at	Index on Faq_Category	This table stores information about each Faqs
Faqs_categories	faq_category_id category_name status created_by created_at updated_by updated_at	Index on None	This table stores information about each Faqs Categories
Financial Years	financial_year_id financial_year_name year_start year_end	Index on None	This table stores information about each Financial Years

	status created_by created_at updated_by updated_at		
Ideas	idea_id financial_year_id theme_problem_id team_id idea_title solution_statement detailed_solution prototype_available Prototype_file idea_available self_declaration status initiated_by submitted_at created_by created_at updated_by updated_at verified_by verified_at evaluated_by evaluated_at evaluation_status rejected_reason final_result district	Index on None	This table stores information about each Ideas
Idea Report	idea_id team_id mentor_id institution_id institution_code institution_name state_name district_name block_name taluk_name place_name principal_name principal_mobile principal_email mentor_name mentor_mobile mentor_email team_name students_names theme_problem_id idea_title solution_statement detailed_solution prototype_available Prototype_file idea_available self_declaration status evaluation_status verified_by overall_score quality_score feasibility_score final_result novelty useful feasibility scalability sustainability	Index on None	This table stores information about each Idea Report

	eval_count		
Institutional Courses	institution_course_id institution_id institution_type_id stream_id program_id status special_category created_by created_at updated_by updated_at	Index on None	This table stores information about each Institutional Courses
Institution	institution_id institution_code institution_name institution_name_vernacular place_id principal_name principal_mobile principal_whatsapp_mobile principal_email status password created_by created_at updated_by updated_at role	Index on None	This table stores information about each Institution
Institution Principle	institution_principal_id institution_id principal_name principal_name_vernacular principal_email principal_mobile ed_cell_coordinator_name ed_cell_coordinator_name_vernacular ed_cell_coordinator_email ed_cell_coordinator_mobile status created_by created_at updated_by updated_at	Index on None	This table stores information about each Institution Principle
Institution report	mentor_id user_id institution_id institution_code institution_name state_name district_name block_name taluk_name place_name principal_name principal_mobile principal_whatsapp_mobile principal_email mentor_title mentor_name mentor_mobile	Index on Institution Id	This table stores information about each Institutaion report

	mentor_whatapp_mobile mentor_email date_of_birth gender team_count student_count submittedcout draftcout		
Institution Types	institution_type_id institution_type institution_short_name sort_order status created_by created_at updated_by updated_at	Index on None	This table stores information about each Institution Types
Instructions	instructions_id slug instructions status created_by created_at updated_by updated_at	Index on None	This table stores information about each Instructions
Latest News	latest_news_id details category url file_name new_status status created_by created_at updated_by updated_at	Index on None	This table stores information about each Latest News
Logins	id username password user_type	Index on None	This table stores information about each Logins
Logs	log_id log_type date message ip method route status_code token headers req_body res_body user_details logged_at	Index on None	This table stores information about each Logs
Mentors	mentor_id financial_year_id user_id institution_id mentor_title mentor_name mentor_name_vernacular mentor_mobile mentor_whatapp_mobile mentor_email date_of_birth gender	Index on None	This table stores information about each Mentors

	otp reg_status status created_by created_at updated_by updated_at		
Mentor Courses	mentor_course_id title description thumbnail status created_by created_at updated_by updated_at	Index on None	This table stores information about each Mentors Courses
New Inst	id old_id place_id	Index on None	This table stores information about each New Inst
Notification	notification_id notification_type target_audience title image message read_by status created_by created_at updated_by updated_at	Index on None	This table stores information about each Notification
Organization	organization_id organization_name organization_code unique_code state district category pin_code city address country status principal_name principal_mobile principal_email created_by created_at updated_by updated_at	Index on None	This table stores information about each Organization
Programs	program_id program_name program_short_name no_of_years program_type sort_order status created_by created_at updated_by updated_at	Index on None	This table stores information about each Program
Quiz	quiz_id no_of_questions status created_by created_at	Index on None	This table stores information about each Quize Programs

	updated_by updated_at		
Resources	esource_id description role type attachments status created_by created_at updated_by updated_at	Index on None	This table stores information about each Course Resources
Students	student_id institution_course_id year_of_study financial_year_id user_id team_id student_full_name date_of_birth mobile email Gender Age certificate_issued status created_by created_at updated_by updated_at	Index on None	This table stores information about Students details
Students_report	student_id mentor_id user_id team_id institution_id institution_course_id institution_code institution_name state_name district_name block_name taluk_name place_name principal_name principal_mobile principal_whatsapp_mobil e principal_email mentor_name mentor_mobile mentor_email year_of_study student_full_name	Index on None	This table stores information about each Students Report

Security Architecture

This section provides details on how security concerns in IT Corp's ERP System are addressed, including encryption, authentication, and authorization strategies.

Encryption

Database At Rest	Data In Transit
The database is encrypted using industry-standard AES-256 encryption. This ensures that all sensitive data stored in the database is secure and unreadable without the proper encryption keys.	All data transmitted between the client and the server is encrypted using TLS (Transport Layer Security). This ensures that sensitive information such as user credentials and personal data is secure as it moves across the network.

Authentication

- **Password Hashing**
User passwords are never stored in plain text in the database. Instead, they are hashed using a strong cryptographic hashing algorithm (e.g., bcrypt) with a unique salt for each user.
- **JWT (JSON Web Tokens)**
After a successful login, the server generates a signed and time-limited JWT, which is sent to the client. The client must include this token in the header of their subsequent requests, allowing the server to identify and validate the user.
- **Authorization Middleware**
Use middleware to enforce role-based access control (RBAC) for API endpoints.
- **API Security**
Input Validation: Implement input validation to prevent injection attacks (e.g., SQL injection, XSS).
Rate Limiting: Apply rate limiting to APIs to mitigate abuse and DoS attacks.
- **Logging and Monitoring**
Logging: Implement logging mechanisms to track and audit API requests, errors, and critical events.
Monitoring: Use monitoring tools to detect anomalies and potential security breaches in real-time.

Environment Arch

- **Development Environment:**
Points to local development servers and URLs, allowing for local testing and development.
- **Production Environment**
Points to live production servers and URLs, ensuring the application operates with production-grade services and URLs.

Development Environment

Overview

In the development environment, the configuration points to local instances or development servers for both the backend API and the landing page URL. This setup allows developers to test and develop the application locally without affecting the production environment. `REACT_APP_LOCAL_LANGUAGE_NAME` and `REACT_APP_LOCAL_LANGUAGE_CODE` variables specify the default language used during development.

```
REACT_APP_NAME = "UNISOLVE"  
REACT_APP_BASE_URL = "http://localhost:8001/"  
REACT_APP_API_BASE_URL = "http://localhost:8001/api/v1"  
REACT_APP_API_IMAGE_BASE_URL = "http://localhost:8001"  
REACT_APP_TEAM_LENGTH = 5  
REACT_APP_USEDICECODE = 1  
REACT_APP_LOCAL_LANGUAGE_NAME = "Tamil"  
REACT_APP_LOCAL_LANGUAGE_CODE = "tn"  
REACT_APP_LANDING_PAGE_URL = "http://localhost:3000/mentor"
```

Production Environment

In the production environment, the configuration points to the live URLs and services that host your application. `REACT_APP_LOCAL_LANGUAGE_NAME` and `REACT_APP_LOCAL_LANGUAGE_CODE` variables specify the default language used in production. This setup ensures that the application interacts with the correct API endpoints and serves content from the appropriate URLs when deployed to production servers.

```
REACT_APP_NAME = "UNISOLVE"  
REACT_APP_BASE_URL = "https://ediitnhackbeta.betaworx.in/api/"  
REACT_APP_API_BASE_URL = "https://ediitnhackbeta.betaworx.in/api/api/v1"  
REACT_APP_API_IMAGE_BASE_URL = "https://ediitnhackbeta.betaworx.in/api/"  
REACT_APP_TEAM_LENGTH = 5  
REACT_APP_USEDICECODE = 1  
REACT_APP_LOCAL_LANGUAGE_NAME = "English"  
REACT_APP_LOCAL_LANGUAGE_CODE = "en"  
REACT_APP_LANDING_PAGE_URL = "https://ediitnhackbeta.betaworx.in/mentor"
```

Dependencies

This section lists all external dependencies, including third-party libraries, APIs, and services that the IT Corp's ERP System relies on.

Dependencies	Packages	Services
Core Dependencies	<ul style="list-style-type: none"> • express • sequelize • jsonwebtoken • bcrypt • axios 	<ul style="list-style-type: none"> • Web framework for building APIs. • ORM for interacting with relational databases. • Library for generating and verifying JSON Web Tokens (JWT). • Hashing passwords for secure storage. • HTTP client for making requests to external APIs.
Middleware and Utilities	<ul style="list-style-type: none"> • helmet: • cors • compression • morgan • dotenv 	<ul style="list-style-type: none"> • Middleware for securing HTTP headers. • Middleware for enabling Cross-Origin Resource Sharing (CORS). • Middleware for compressing HTTP responses. • HTTP request logger middleware for Node.js. • Module for loading environment variables from .env files.
Testing and Development	<ul style="list-style-type: none"> • jest • supertest • nodemon: • eslint 	<ul style="list-style-type: none"> • Testing framework for JavaScript projects. • Library for testing HTTP servers. • Utility for automatically restarting the server during development. • Tool for identifying and reporting on patterns found in JavaScript code.
Database and ORM	<ul style="list-style-type: none"> • mysql2 • sequelize-cli 	<ul style="list-style-type: none"> • MySQL client with promise support. • Command-line interface for Sequelize migrations and seeders.
Additional Utilities	<ul style="list-style-type: none"> • lodash • dayjs • uuid: 	<ul style="list-style-type: none"> • Utility library providing helper functions. • Library for manipulating and formatting dates.

	<ul style="list-style-type: none"> • dotenv-safe: 	<ul style="list-style-type: none"> • Library for generating unique identifiers. • Securely loads environment variables from .env files.
Security and Authentication	<ul style="list-style-type: none"> • passport: • express-session 	<ul style="list-style-type: none"> • Authentication middleware for Node.js. • Session middleware for Express.js applications.
Other Utilities	<ul style="list-style-type: none"> • axios: 	<ul style="list-style-type: none"> • Promise based HTTP client for the browser and node.js

Configuration Management

This section explains how system configurations are managed and how they can be changed.

1. **Ansible** is used to automate the process of configuration management across all servers and environments.
 2. Critical configurations such as database connections and API keys are managed using environment variables, ensuring they are not hard-coded into the application.
 3. Any changes to system configurations are documented and go through a change approval process involving the DevOps team and relevant stakeholders.
-



Support

If you encounter any issues, have questions, or need assistance with the project, please don't hesitate to reach out to our support team. We are here to help you get the most out of our software.

Contact Information

- Email : <mailto:info@helloindiasolutions.com>
- Phone : +91 9840441395 , +91 9841316211
- Website : <https://www.helloindiasolutions.com/>

Thank You!

We would like to extend our heartfelt gratitude for choosing our software. Your support and feedback are invaluable to us as we continue to improve and innovate.