Documento de Projeto de Sistema

English For All Time

Vitória, ES

2025

Registro de Alterações:

Versão	Responsável	Data	Alterações
1.0	Matheus De Oliveira	27/05/2025	Versão inicial.

1 Introdução

Este documento apresenta o projeto (*design*) do sistema *English For All Time*.

É um plataforma de ensino de inglês onde o professor-administrador tem controle total: ele pode cadastrar seus alunos e adicionar/criar cursos diretamente no sistema, organizando-os em módulos com vídeos (via links do YouTube não listados), materiais em PDF e exercícios. A plataforma oferece um painel intuitivo para o dono gerenciar tanto os usuários quanto os conteúdos publicados, permitindo atualizações rápidas e personalizadas, sem depender de terceiros. E para os alunos eles terão uma página com todos os conteúdos publicados pelo professor.

Além desta introdução, este documento está organizado da seguinte forma: a Seção 2 apresenta a plataforma de software utilizada na implementação do sistema; a Seção **??** apresenta a especificação dos requisitos não funcionais (atributos de qualidade), definindo as táticas e o tratamento a serem dados aos atributos de qualidade considerados condutores da arquitetura; a Seção 3 apresenta a arquitetura de software; por fim, a Seção 4 apresenta os modelos FrameWeb que descrevem os componentes da arquitetura.

2 Plataforma de Desenvolvimento

Na Tabela 1 são listadas as tecnologias utilizadas no desenvolvimento da ferramenta, bem como o propósito de sua utilização.

Tabela 1 – Plataforma de Desenvolvimento e Tecnologias Utilizadas.

Tecnologia	**Versão**	**Descrição**	**Propósito**
React.js	18+	Biblioteca JavaScript para interfaces dinâmicas	Frontend responsivo para alunos e admin
TypeScript	5.x	Superset tipado de JavaScript	Frontend responsivo para alunos e admin
Spring Boot	3.2.x (Java 17)	Framework backend Java	API RESTful segura e escalável
Spring Web MVC	6.1.x	Módulo para construção de APIs REST	Rotas HTTP e serialização JSON
Spring Security	6.1.x	Autenticação e autorização	Controle de acesso (JWT)
Spring Data JPA	3.1.x	Persistência com Hibernate	Operações de banco de dados (PostgreSQL)
PostgreSQL	15+	Banco de dados relacional	Armazenar usuários, cursos e progresso
jjwt	0.12.x	Biblioteca para JWT	Geração/validação de tokens
Lombok	1.18.x	Redução de boilerplate em classes Java	Getters/Setters automáticos

Tecnologia	Versão	Descrição	Propósito
Hibernate Validator	8.0.x	Validação de dados em DTOs	Validar entradas de API (ex.: @Email, @NotBlank)
React Router	6.x	Roteamento no frontend	Navegação entre páginas
Axios	1.x	Cliente HTTP para frontend	Consumir API do backend
Material UI	5.x	Biblioteca de componentes UI	Design consistente e responsivo

Na Tabela 2 vemos os softwares que apoiaram o desenvolvimento de documentos e também do código fonte.

Tabela 2 – Softwares de Apoio ao Desenvolvimento do Projeto

Tecnologia	Versão	Descrição	Propósito
PgAdmin 4	7.x+	Interface gráfica para PostgreSQL	Gerenciar visualmente o banco de dados
IntelliJ IDEA	2023.2+	IDE para Java/Spring Boot	Desenvolvimento backend com debug integrado
VS Code	1.80+	Editor para React/TypeScript	Codificação do frontend com extensões úteis
Postman	10+	Teste de APIs	Validar endpoints do Spring Boot
Git	2.40+	Controle de versão	Gerenciar colaboração no código
Visual Paradigm	17.2	Criação de diagramas UML	Criação dos modelos de Entidades, Aplicação, Persistência e Navegação.
FrameWeb	-	Plugin do Visual Paradigm	Auxilia na criação da estrutura FrameWeb para o desenvolvimento dos modelos.
TeX Live	2018	Implementação do LaTeX	Documentação do projeto arquitetural do sistema.
TeXstudio	4.8.7	Editor de LaTeX.	Escrita da documentação do sistema.
Apache Maven	3.5	Ferramenta de gerência/construção de projetos de software.	Obtenção e integração das dependências do projeto.

3 Arquitetura de Software

A Figura 1 mostra a arquitetura do sistema *English For All Time*. Ela é baseada em uma combinação dos estilos arquitetônicos Camadas e Partições.

Cada partição é, então, subdividida em três camadas: (1) *Camada de Interface com o Usuário* (CIU), responsável pela interação com os usuários, exibindo dados e capturando as ações externas ao sistema; (2) *Camada de Lógica de Negócio* (CLN), responsável pela representação dos elementos do domínio e implementação das funcionalidades do sistema;

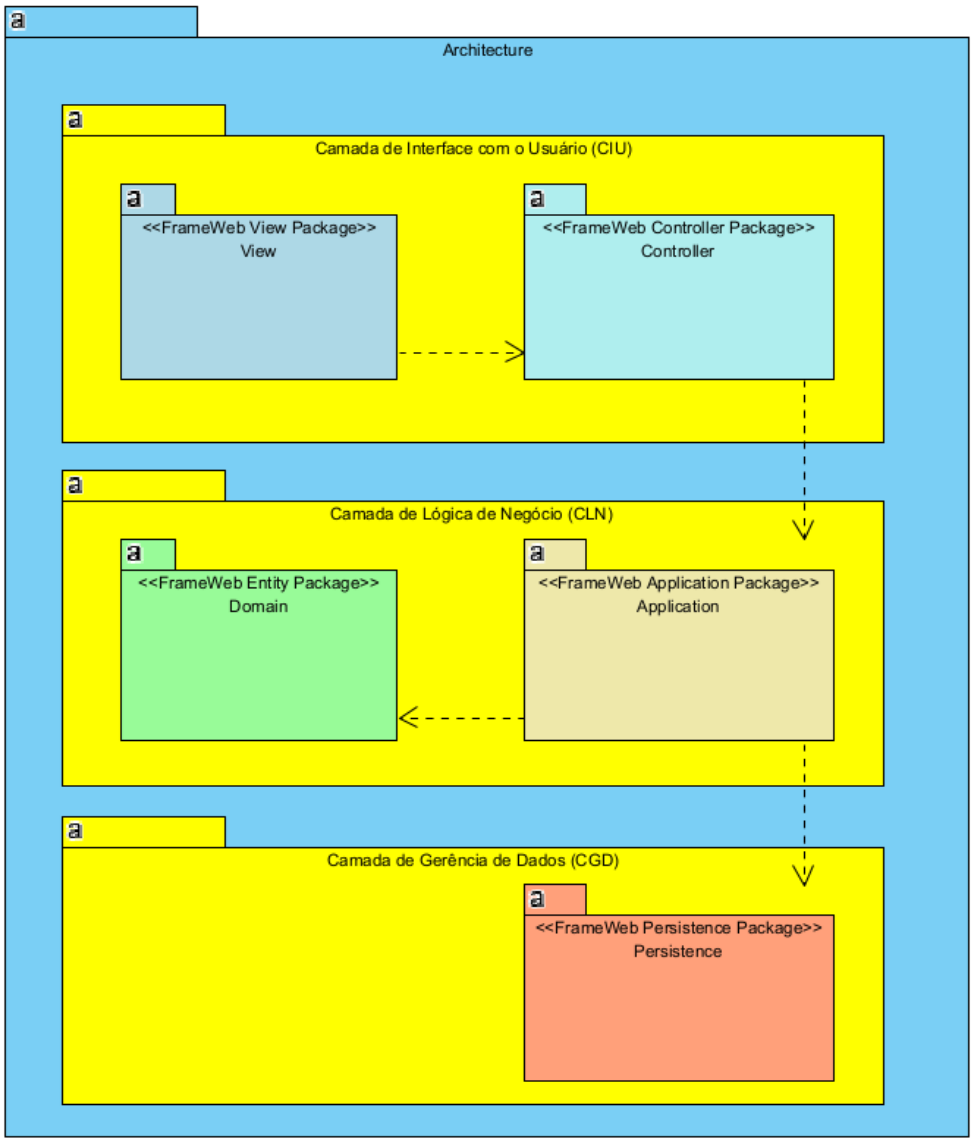

Figura 1 – Arquitetura de Software.

(3) *Camada de Gerência de Dados* (CGD), responsável pela persistência dos objetos em banco de dados relacionais.

Na CIU, adota-se o padrão Modelo-Visão-Controlador-Service (MVCS), uma extensão do tradicional Modelo-Visão-Controlador (MVC), adptado à arquitetura moderna de aplicações Web, dividindo-a nos pacotes: view (visão), que agrupa as páginas Web e demais elementos da camada de apresentação, como folhas de estilo, imagens e scripts de cliente; controller (controle), que contém as classes controladoras responsáveis por coordenar a interação entre a interface do usuário (visão) e os serviços da aplicação, localizados na Camada Lógica de Negócio (CLN). A visão depende do controle de forma unidirecional, assim como o controle depende unicamente da camada de aplicação, preservando a independência da lógica de negócio em relação à interface com o usuário.

Na CLN, aplica-se o padrão Camada de Serviço (FOWLER, 2002), correspondendo

ao componente "Service"do padrão MVCS. A camada é organizada em dois pacotes principais: domain (domínio), que contém as classes que representam os elementos centrais do domínio do problema; application (aplicação), que contém classes que implementam as funcionalidades do sistema. Para implementar suas funções, a aplicação depende do domínio, pois manipula seus objetos, e da persistência, para armazená-los no banco de dados.

Na CGD, aplica-se o padrão Objeto de Acesso a Dados (Data Acess Object ou DAO) (**??**), contendo um único pacote, persistence (persistência), cuja classes são responsáveis pelas operações de persistência (utilizando mapeamento objeto/relacional) de uma única classe de domínio cada.

4 Modelagem FrameWeb

English For All Time é um sistema Web cuja arquitetura utiliza *frameworks* comuns no desenvolvimento para esta plataforma. Desta forma, o sistema pode ser modelado utilizando a abordagem FrameWeb (SOUZA, 2020).

A Tabela 3 indica os *frameworks* presentes na arquitetura do sistema que se encaixam em cada uma das categorias de *frameworks* que FrameWeb dá suporte. Em seguida, os modelos FrameWeb são apresentados para cada camada da arquitetura.

Vítor: Substituir os valores da segunda coluna da Tabela 3 pelos *frameworks* utilizados no seu projeto. Remover o fundo amarelo.

Tabela 3 – *Frameworks* da arquitetura do sistema separados por categoria.

Categoria de *Framework*	***Framework* Utilizado**
Controlador Frontal	JSF
Injeção de Dependências	CDI
Mapeamento Objeto/Relacional	JPA
Segurança	JAAS

4.1 Camada de Negócio

Vítor: Apresentar os modelos de entidades e de aplicação do FrameWeb.

4.2 Camada de Acesso a Dados

Vítor: Apresentar os modelos de persistência do FrameWeb.

4.3 Camada de Apresentação

Vítor: Apresentar os modelos de navegação do FrameWeb.

Referências

FOWLER, M. *Patterns of Enterprise Application Architecture*. 1. ed. [S.l.]: Addison-Wesley, 2002. ISBN 9780321127426. Citado na página 4.

SOUZA, V. E. S. The FrameWeb Approach to Web Engineering: Past, Present and Future. In: ALMEIDA, J. P. A.; GUIZZARDI, G. (Ed.). *Engineering Ontologies and Ontologies for Engineering*. 1. ed. Vitória, ES, Brazil: NEMO, 2020. cap. 8, p. 100–124. ISBN 9781393963035. Disponível em: <http://purl.org/nemo/celebratingfalbo>. Citado na página 5.