**Etapa 3 – Gestão de Configuração de Software**

Nesta etapa, iremos organizar e estruturar a configuração de nosso sistema de modo que possamos dividir nossa proposta em várias camadas de organização, levando em consideração a arquitetura do sistema, os componentes principais e as funcionalidades que deverão ser oferecidas. Sendo assim, sugerimos uma proposta de arquitetura de sistema, considerando uma abordagem modular, escalável e com foco na segurança e usabilidade.

**1. Arquitetura de Sistema**

Nossa arquitetura pode ser baseada em uma **arquitetura de microsserviços** ou em uma **arquitetura monolítica** dependendo da escala e necessidade do sistema. Para efeitos de uma arquitetura mais flexível e com capacidade para futuras expansões, vamos considerar uma arquitetura de microsserviços.

**Camadas principais do sistema:**

* **Frontend (Interface do Usuário)**:

**Plataforma web e/ou app**: Será responsável pela interação do aluno com o sistema. Pode ser acessado via navegador (web) ou como um aplicativo mobile (Android/iOS). Poderemos utilizar de tecnologias como React, Angular ou Vue.js para web; React Native ou Flutter para mobile para essa finalidade.

* **Backend (Servidor de Lógica de Negócio)**:

**API Restful ou GraphQL**: Para comunicação entre frontend e backend, como tecnologias podemos utilizar Node.js, Django, Ruby, Spring Boot ou .NET.

**Segurança**: Autenticação baseada em **OAuth2** ou **JWT (JSON Web Token)** para gerenciamento de sessões e segurança no login.

* **Banco de Dados**:

**Banco de Dados Relacional** (SQL): como nosso sistema exige muitas consultas complexas sobre dados estruturados (notas, matrículas, históricos etc.), um banco de dados relacional é o ideal, podemos nos favorecer de tecnologias como PostgreSQL ou MySQL.

* **Armazenamento de Documentos**:

**Serviço de Armazenamento de Arquivos**: como nosso sistema permitirá o envio de documentos, é preciso integrar uma solução de armazenamento de arquivos (por exemplo, para envio de trabalhos). Podemos utilizar os serviços da Amazon S3, Google Cloud Storage ou armazenamento local para essa finalidade.

* **Mensagens**:
  + **Sistema de Notificações**: Para alertar os alunos sobre matrículas, prazos, novas notas ou atividades. Firebase Cloud Messaging, RabbitMQ são algumas das opções disponíveis para esse serviço.

**2. Modelagem de Dados**

A modelagem de dados precisa ser bem estruturada para garantir que o sistema consiga armazenar e recuperar as informações com eficiência. Sendo assim, algumas tabelas principais para o sistema poderiam ser:

**Entidades principais:**

* **Aluno**:

RA (identificador único)

Nome

Senha (armazenada de forma segura, utilizando hashing)

E-mail

Telefone

Foto (opcional)

Dados pessoais

* **Matéria**:

Código

Nome

Professor

Semestre

Horários de aula

Datas de provas

* **Matrícula**:

RA (relacionado com o Aluno)

Código da Matéria

Status da Matrícula (confirmada, pendente, rematrícula)

* **Notas**:

RA (relacionado com o Aluno)

Código da Matéria

Nota 1 (Prova 1)

Nota 2 (Prova 2)

Nota Final

Frequência

Status de aprovação (Aprovado, Reprovado)

* **Presença**:

RA (relacionado com o Aluno)

Código da Matéria

Data da Aula

Presença (Sim/Não)

* **Documentos**:

RA (relacionado com o Aluno)

Tipo de Documento (por exemplo, atestado, comprovante de matrícula)

Arquivo (link para o arquivo armazenado)

Data de envio

Status (Aprovado/Pendente)

**3. Funcionalidades do Sistema**

Para implementar a funcionalidade do sistema, podemos dividir as funções e requisitos principais que ele deve oferecer:

**Autenticação e Autorização:**

* O aluno realiza o login utilizando o RA e a senha.
* A senha deve ser armazenada de maneira segura utilizando hashing, como por exemplo bcrypt, scrypt.
* O sistema deve validar os dados e garantir que a autenticação seja bem-sucedida antes de permitir o acesso a qualquer funcionalidade.
* O sistema deve ter sessões e tokens JWT para gerenciar a autenticação de forma eficiente.

**Funcionalidades para o Aluno:**

1. **Consultar Notas e Presença**:

O aluno pode acessar suas notas de cada matéria.

O aluno pode consultar seu histórico de presença/falta em cada aula.

1. **Matrícula e Rematrícula**:

O aluno pode realizar a matrícula ou rematrícula em matérias, conforme o período.

O sistema deve mostrar quais matérias o aluno pode se matricular com base na grade curricular e pré-requisitos.

1. **Consultar Grade Curricular**:

O aluno pode consultar a grade curricular do curso.

O sistema pode listar as matérias obrigatórias e optativas de cada semestre.

1. **Consultar Histórico de Notas**:

O aluno pode visualizar o histórico completo de suas notas ao longo dos semestres.

1. **Horários de Aula e Avaliações**:

O aluno pode acessar os horários das aulas e datas de provas/trabalhos.

1. **Solicitar e Enviar Documentos**:

O aluno pode enviar documentos como atestados, comprovantes de matrícula, etc.

O sistema pode permitir que os alunos solicitem documentos específicos, como históricos acadêmicos, por exemplo.

**4. Fluxos de Interação do Sistema**

**Fluxo de Login:**

1. O aluno insere seu RA e senha.
2. O sistema valida a autenticidade dos dados.
3. Se o login for bem-sucedido, o aluno é redirecionado para o painel de controle.
4. Se falhar, o sistema deve mostrar uma mensagem de erro clara.

**Fluxo de Matrícula:**

1. O aluno acessa a área de matrícula.
2. O sistema apresenta uma lista das matérias disponíveis para matrícula.
3. O aluno escolhe as matérias e realiza a matrícula.
4. O sistema verifica se o aluno pode se matricular nas matérias (verificando pré-requisitos, por exemplo).
5. O sistema atualiza o banco de dados com as matérias matriculadas.

**5. Segurança e Privacidade**

A segurança do sistema é crucial, especialmente ao lidar com dados pessoais e acadêmicos. Algumas práticas importantes são:

* **Autenticação e autorização**: podemos utilizar JWT e OAuth2 para gerenciar sessões de login.
* **Proteção contra ataques comuns**: implantar proteção contra SQL Injection, XSS e CSRF são boas opções disponíveis para segurança.
* **Criptografia**: Armazenar senhas de maneira segura utilizando algoritmos de hashing fortes (bcrypt).
* **Controle de acesso**: Restringir o acesso às informações de acordo com o tipo de usuário (aluno, professor, administrador).

1. **Escalabilidade e Manutenção**

Por fim, devemos nos certificar de que o sistema seja escalável e fácil de manter, de acordo com os seguintes dados:

* **Microsserviços**: caso o sistema cresça, nossa arquitetura de microsserviços permitirá dividir o sistema em serviços independentes, como autenticação, matrícula, notas etc, garantindo assim uma maior flexibilidade e facilidade na manutenção.
* **CI/CD**: o implante uma pipeline de integração e entrega contínua deverá garantir que as atualizações possam ser feitas de maneira eficiente e sem interrupções ao longo do processo.
* **Monitoramento e Log**: utilização de ferramentas como Prometheus, Grafana e ELK stack para monitoramento e logs.