ANEXO 1

1. Informações Iniciais

Informações	
Nome do projeto	HOMIN+
Data de início	29/08/2025
Cliente	UNINASSAU
Envolvidos	Arthur Vieira Pinto 16032764 Danny Kevelyn Santos - 16033292 Ian Matheus Zambanini - 16036573 Jailson Nascimento Dos Santos Sena - 16033795 José Pereira Dos S Neto -16032565 Maria Vitória da Silva - 16033768 Mariana dos Santos Andrade - 16033830 Maycon Felipe dos Santos - 16033023 Priscila Sousa Mota -16032642 Reno Soares dos Santos - 16033002
Gerente de Projeto	Danny Kevelyn Santos - 16033292

2. Histórico de versão do documento

Versão	Descrição	Autor(es)
1.0	Preenchimento Inicial da documentação	Priscila Sousa Mota Danny Kevelyn
1.1	Adição de backlog do produto e cronograma inicial	Maycon Felipe

3. Do Projeto

Informações	
Conceito	O Homiin+ está sendo desenvolvido de forma colaborativa e interdisciplinar, com foco em entregar um MVP funcional e validado que una tecnologia acessível, design intuitivo e conteúdo confiável voltado à saúde do homem. O objetivo é criar uma plataforma que incentive o cuidado preventivo, fornecendo informações verificadas por especialistas e suporte inteligente por meio de inteligência artificial (IA) integrada.
Tecnologia	A aplicação utiliza React.js no front-end, priorizando acessibilidade e usabilidade, e FastAPI (Python) no back-end, garantindo desempenho e escalabilidade. A arquitetura integra LangChain, Agno, ChromaDB e OpenAI API, formando um sistema de IA baseado em RAG (Retrieval-Augmented Generation) que combina busca vetorial local com consulta em fontes confiáveis da web.O banco de dados é híbrido, com SQL relacional para usuários e logs, e ChromaDB para embeddings semânticos.

Link da aplicação online	
Link do projeto (código) no GitHub	https://github.com/mayconfp/Homin_Backend https://github.com/mayconfp/Homin_frontend
Link de pasta com vídeo da aplicação.	https://drive.google.com/file/d/13wKBl63OZsgZoYguTEQ1pKINHIx-tuAZ/view?usp=sharing
Público- alvo	Homens adultos, entre 18 e 50 anos, interessados em adotar hábitos de autocuidado, prevenção de doenças e bem-estar físico e mental. O projeto também visa impactar positivamente instituições e profissionais da área da saúde que buscam novas formas de engajar o público masculino.

4. Metodologia de Desenvolvimento

Ação	Descrição
Nome da Metodologia	Scrum
Data início de aplicação *Sprint*	29/08/2025
Data fim de aplicação da *Sprint*	19/09/2025
Quantas semanas (*Sprint*)	3 semanas
Product Owner	Priscila Sousa Mota
Scrum Master	Maycon Felipe dos Santos
Ferramenta de gestão	Jira

5. Principais artefatos e Eventos

Backlog do Produto	
Código	**Descrição**
BP01 – Criação do protótipo	Criação do protótipo navegável no Figma, incluindo identidade visual, fluxos de navegação, telas principais e responsividade.
BP02 – Estruturação da comunicação e marketing digital	Criação do perfil oficial no Instagram (@homiin.saude), identidade visual para redes e planejamento de conteúdo.
BP03 – Planejamento técnico	Definição da arquitetura geral do sistema, tecnologias (React.js, FastAPI, LangChain, Agno e ChromaDB) e padronização do repositório no GitHub.
BP04 – Desenvolvimento do front-end	Implementação da interface em React.js com cards interativos, autenticação e layout responsivo.
BP05– Desenvolvimento do back-end	Criação dos módulos de usuários, documentos e logs, além da integração com o motor de IA (RAG).
BP06– Integração da IA Engine	Integração da IA Engine para respostas inteligentes baseadas em conhecimento local e fontes confiáveis da web.
BP07– Testes e validação	Testes de usabilidade, correção de bugs e análise de desempenho da IA.
BP08– Entrega final e documentação:	Gravação de vídeo demonstrativo, preenchimento dos relatórios e submissão à UNINASSAU.

Data da reunião para *backlog*:

29/08/2025

Planejamento *Sprint*	
Ação	Durante o planejamento da Sprint 1, a equipe definiu as tarefas prioritárias para a construção do MVP, com foco na prototipagem do sistema, criação da identidade visual e estruturação técnica do projeto. O backlog foi analisado e dividido entre os integrantes conforme suas habilidades, garantindo que as entregas fossem viáveis dentro do prazo de três semanas.
Data da Sprint (inicio/fim)	29/08/25 - 19/09/25
Como aconteceu a definição das ações?	As ações foram definidas em uma reunião inicial, onde a equipe revisou os requisitos e dividiu as responsabilidades de acordo com as áreas (Design, IA, Front-end, Back-end e DBA). O Jira foi utilizado para registrar as tarefas e acompanhar o progresso de cada membro.

Reunião Diária	
Ação	As reuniões foram realizadas via Discord e WhatsApp, com duração de aproximadamente 15 minutos. Cada integrante compartilhou o que havia feito no dia anterior, o que faria em seguida e possíveis bloqueios encontrados. Essa prática ajudou a manter o alinhamento entre as áreas de design, desenvolvimento e conteúdo.
RD01 – Semana 1	As reuniões iniciais focaram na divisão de tarefas e definição das responsabilidades. A equipe de design apresentou os primeiros rascunhos do protótipo no Figma e discutiu ajustes na identidade visual.
RD02 – Semana 2	Os integrantes de marketing e comunicação criaram o perfil no Instagram e iniciaram o planejamento de postagens. O time técnico definiu a arquitetura da aplicação e revisou as tecnologias que seriam utilizadas no MVP.
RD03 – Semana 3	Foi concluído o protótipo navegável e realizada a validação visual entre os membros. Também foram definidos os próximos passos

	para o início do desenvolvimento do front-end e documentadas as decisões técnicas no Jira.

Revisão da Sprint	
Ação	**Descrição**
RS01 - Revisão	Foram concluídas as principais definições conceituais e estruturais do projeto Homin+, incluindo os diagramas (DER, UML, Caso de Uso, Classe e Fluxograma), a prototipagem inicial no Figma, a definição da arquitetura técnica e os requisitos funcionais e não funcionais. Todas as tarefas foram registradas e controladas pelo Jira.
RS02- Revisão	A equipe validou o fluxo geral do sistema (login, chat IA e administração de documentos) e a estrutura de diretórios do back-end e front-end. O protótipo foi aprovado como base para o desenvolvimento da próxima sprint, que focará na implementação prática.

Retrospectiva da Sprint	
Ação	**Descrição**
RTS01 - Revisão	A equipe avaliou que a organização das tarefas no Jira facilitou o acompanhamento das atividades. Houve boa comunicação entre os membros, mas identificou-se a necessidade de melhorar a integração entre as áreas de IA e front-end. Para a próxima sprint, será reforçada a definição de prazos intermediários e revisões semanais.

6. Cronograma

Cronograma			
Data	**Descrição**	**Realizador**	**Status**
29/08/2025	Inicialização do projeto	Equipe	Concluído
02/09/2025	Prototipagem no Figma e identidade visual .	Priscila Mota	Andamento
03/09/2025	Definição de documentos e para Rag .	Maycon Felipe	Concluído
05/09/2025	Definição das tabelas DB	Reno Soares	Concluído
07/09/2025	Diagrama caso de uso	Danny Kevelyn	Concluído
08/09/2025	Diagrama UML	Mariana Andrade	Concluído
09/19/2025	Diagrama de Classe	José Neto	Concluído
10/09/2025	Criação perfil Instagram @homiin.saude Marketing	Arthur Vieira	Concluído
11/09/2025	Definição de requisitos	Maycon e Priscila	Concluído

16/09/2025	Diagrama de estado	Jailson Nascimento	Concluído
18/09/2025	Atualização das mídias sociais	Arthur Vieira	Andamento
19//09/2025	Finalização da Sprint	Equipe	Concluído
08/10/2025	Entrega de Documentação	Equipe	Concluído
08/10/2025	Criação de repositório	Maycon Felipe	Concluído
10/10/2025	Testes e Validações	Ian Zambanini	Á fazer

6. Requisitos Funcionais (RF)

Código	Descrição
RF01 -	Sistema de autenticação múltipla (Google, Auth0, Email/Senha)
RF02 -	Chat inteligente com IA para perguntas sobre saúde masculina.
RF03-	Busca em base de conhecimento local e web em tempo real.
RF04 -	Busca automática em fontes confiáveis ,quando não for encontrada na base local.
RF05-	Armazenamento de documentos pelo Administradores(PDF)
RF06-	Registro e listagem de documentos disponíveis na base de conhecimento.
RF07-	Registro de logs de conversas entre usuário e a IA.
RF08-	Exibição de respostas ,claras e contextualizada ao Usuário
RF09-	Diferenciação de acesso entre Usuários ,comuns e Adms.
RF10-	Acesso restrito a IA por meio de Login .

7. Requisitos Não Funcionais (RNF)

Código	Descrição
RNF01 -	Interface Responsiva
RNF02 -	Tempo resposta de IA <10 segundos exceto em busca na internet.
RNF03-	Comunicação via protocolo HTTPS .(Cliente e Servidor e APIs)
RNF04-	O sistema deve garantir a diferenciação de papeis entre Usuário e Administrador .
RNF05-	Código deve ser versionado e armazenado no GitHUB
RNF06-	O sistema deve manter 95% de disponibilidade mínima durante o uso.
RNF07-	A base de conhecimento deve ser atualizada sem necessidade de interrupções do sistema .
RNF08-	O sistema deve registrar logs de erro para monitoramento.
RNF09-	Interface será desenvolvida React.js
RNF10-	As credenciais de acesso devem ser armazenadas de forma segura e criptografada .

8. Avaliação da Interação

Como foi feita a avaliação da interação
Resultados da avaliação da interação
Sugestões de mudança das interfaces a partir dos resultados da avaliação da interação

9. Relatório

Como a equipe dividiu as tarefas para desenvolver e gerar as versões da aplicação?
Arthur Vieira Pinto – Mídias Sociais e Marketing Danny Kevelyn Santos – Front-end/ documentação Ian Matheus Zambanini – Back-end Jailson Nascimento dos Santos Sena – Front-end e Qa José Pereira dos Santos Neto – Front-end Maria Vitória da Silva – Front-end - Testes e Validaçoes Mariana dos Santos Andrade – Front-end Maycon Felipe dos Santos – Back-end Priscila Sousa Mota – UX / Design e a prototipação Reno Soares dos Santos – DBA
Fluxograma do usuário
Acesso inicial O usuário abre a aplicação Homiin+ no navegador. A interface é exibida (conteúdos, cards de dicas, artigos, etc.). Na UI existe um ícone/botão da IA; esse ícone: Está visível, mas ao clicar verifica se o usuário está autenticado; Se o usuário não estiver autenticado, é exibida a página de login (porque só é permitido falar com a IA quando autenticado). Se já autenticado, abre o painel/caixa de interação com a IA. Página de login (quando o usuário decide usar a IA) O usuário é redirecionado para a Página de Login. Ele escolhe o método de autenticação:

Login Google → GoogleAuth.LoginGoogle.

Login Email/Senha → EmailSenhaAuth.LoginComSenha.

O serviço de autenticação valida credenciais e retorna resultado.

Decisão: Autenticação bem sucedida?
Se não (falha):

Exibir erro/alerta na página de login e permitir nova tentativa.

Se sim:

Criar sessão / token de acesso e redirecionar o usuário para a interface com privilégios (acesso ao uso da IA e demais áreas autenticadas).

Registrar (opcional) o login em logs.

Acesso ao sistema (após login) — visão geral das opções

Usuário autenticado tem acesso total à área que permite:

Conversar com a IA
Para administradores: operações de gestão de documentos (adicionar, listar, buscar).

Usuário não autenticado continua podendo navegar pelo conteúdo público, mas não recebe respostas da IA.

Fluxo de pergunta/resposta com a IA (RAG)

Usuário autenticado faz uma pergunta pela interface da IA.

O front-end chama o serviço responsável: RAGService.responderPergunta (ou equivalente).

Inicia-se a etapa Busca Informação (decisão/processo de procurar contexto).

Prioridade de busca (comportamento esperado)

Primeira tentativa: buscar em fontes internas:

Chamar RepositórioDocumento.buscar (consulta semântica na base interna — por exemplo: ChromaDB).

Avaliar relevância / confiança dos documentos retornados.

Se a base interna contém contexto suficiente (alta relevância/confiança):

Pular busca externa e usar apenas resultados internos.

Se a base interna NÃO trouxer contexto suficiente (nenhum resultado relevante ou confiança baixa):

Acionar BuscaWebAPI.buscar para consultar fontes confiáveis na web (via LangChain/Agno).

(Opcional) Pode-se executar ambas buscas em paralelo e priorizar internas — mas regra principal: priorizar base interna, web só como fallback.

Processamento e geração de resposta

Recebidos os resultados (internos e/ou web), o sistema Processa Resultados:

Normaliza, compila e ordena evidências.

Verifica citações, marca fontes (quando vier da web) e calcula confiança.

RAGService (ou componente de geração) Gera Resposta usando o modelo de linguagem, incorporando o contexto recuperado.

A resposta final é formatada (inclui possíveis referências/fonte quando a informação veio da web).

Exibir a resposta ao usuário na interface.

Registrar a interação em logs_assistant (pergunta, resposta, fontes usadas, timestamp, id do usuário).

Perguntar ao usuário: "Deseja nova pergunta?"

Decisão: Nova pergunta?
Se sim, voltar ao passo usuário faz pergunta pela interface (repetir ciclo de busca/resposta).
Se não, encerrar a interação (manter sessão ativa até logout).

Operações administrativas sobre documentos (usuários com perfil Admin)

Usuários com permissão de Administrador veem/realizam operações extras no sistema:

Adicionar Documento → chama RepositórioDocumento.salvar:

Upload do PDF/arquivo, extração de texto, validação, extração de metadados, vetorização e indexação na ChromaDB; gravação de metadados no SQL.

Listar Documentos → chama RepositórioDocumento.listar:

Retorna lista paginada com metadados.

Buscar Documento (admin) → chama RepositórioDocumento.buscar:

Permite busca interna com filtros, revisões e re-indexações se necessário.

Cada operação de alteração gera logs e (se necessário) notificações para revisão/validação.

Encerramento / Logout / Falhas

O usuário pode fazer logout a qualquer momento:

Token/sessão são invalidados.

Após logout, acesso à IA é bloqueado; conteúdo público permanece acessível.

Em caso de falha na busca web (API externa indisponível):

Sistema deve retornar resposta informando limitação e/ou usar somente as fontes internas disponíveis.

Registrar erro em logs e, se necessário, acionar retry/backoff.

Em caso de falha ao adicionar documento:

Mostrar erro ao administrador com detalhes; permitir reenvio ou rollback.

Exemplo de dois fluxos simples

Exemplo A — Navegação sem login

Usuário entra no site → vê artigos e cards → lê conteúdo → sai.
Resultado: nenhuma autenticação, sem acesso à IA.

Exemplo B — Usar a IA

Usuário clica no ícone da IA → é solicitado login → escolhe Google → autentica com sucesso → retorna à UI da IA.

Faz pergunta → RAGService.responderPergunta → consulta interna (RepositórioDocumento.buscar) → se preciso, consulta web (BuscaWebAPI.buscar) → processa resultados → gera e exibe resposta → log da interação.

Usuário faz nova pergunta ou encerra.

Wireframes ou mockups

https://www.figma.com/design/5IKyZE8Btg0ECGg2JoAEMz/MVP---MEN--S-HEALTH--HOMIN--?node-id=1-2&t=nYMkBDmz2430w88F-1