BETTER PERGAMUM

O **Better Pergamum** é um sistema de biblioteca desenvolvido em C++ que replica funcionalidades do sistema Pergamum da UFV, com adição de recursos modernos como fóruns de discussão por livro, autenticação integrada e interface baseada em sockets.
O sistema foi feito como trabalho final da disciplina INF112, ministrada pelo professor Julio Cesar Soares dos Reis, no curso Ciências da Computação da Universidade Federal de Viçosa

O sistema é composto por dois componentes principais:

- Servidor: Gerencia conexões, fóruns de discussão e armazenamento de mensagens.
- Cliente: Interface para usuários (alunos e professores) acessarem funcionalidades da biblioteca.

Pré-requisitos:
Dependências manuais:
- g++ (compilador C++)
- libcurl4-openssl-dev
- libsqlite3-dev
- pthreads (incluído no glibc)

O sistema funciona apenas no Linux, pois foi feito para tal.

Instalação manual
sudo apt-get update
sudo apt-get install build-essential libcurl4-openssl-dev libsqlite3-dev

Compilação:
Compila servidor e cliente
make all
make clean (remove executáveis gerados)

Método alternativo (sem Makefile):
g++ server.cpp aluno.cpp main_server.cpp client.cpp usuario.cpp livro.cpp chat.cpp biblioteca.cpp mensagem/database.cpp -pthread -lcurl -lsqlite3 -o server; ./server
g++ client.cpp main_client.cpp professor.cpp usuario.cpp livro.cpp aluno.cpp -pthread -lcurl -o client; ./client

Execução:
./server
(o servidor iniciará na porta 12345)

./client

(interface ativa será iniciada)

⚠ Atenção!
O server **precisa** estar aberto para o client funcionar. O usuário pode abrir varios clients para interagir um com o outro

Breve tutorial de uso:
Para usuários (clientes)

Execução:
./client

Seleção do tipo de usuário:
-------SEJA BEM VINDO AO BETTER PERGAMUM-------
Você é:
1 - Aluno
2 - Professor
Resposta: _

Autenticação:
PARA ALUNOS:
Digite sua matricula: [sua_matricula]
Digite a sua senha da BBT: [sua_senha]

PARA PROFESSORES:
Digite seu Nome: [nome_completo]
Digite o seu email: [email_institucional]

Menu dos alunos:
Escolha a função que você deseja executar:
1 - Pesquisa de livros
2 - Consultar o débito
3 - Visualizar perfil
4 - Encerrar programa
Resposta: _

Menu dos professores:
Escolha a função você deseja executar:
1 - Pesquisa de livros
2 - Visualizar perfil
3 - Encerrar programa
Resposta: _

Busca de livros (digitando 1):

PESQUISA DE LIVROS:

Digite os termos para a pesquisa: [termo_de_busca]

Primeiro resultado:

| Nome: [Título do Livro]
| N.Chamada: [Número de Chamada]

Deseja acessar o forum do livro?
Se sim, digite 1, Caso contrário, digite qualquer outro número: _

Acesso ao fórum (chat) (pressionando 1 novamente): – cada livro possui seu próprio chat, um espaço de convívio e compartilhamento de conhecimento acerca do livro

Digite seu username para entrar no fórum: [seu_nickname]

Exemplo de fórum:
FORUM DO LIVRO: [Número de Chamada]
[Data] [Nome do Usuário]: [Mensagem anterior]
[Data] [Nome do Usuário]: [Mensagem anterior]
...
Você está online, digite quit para sair~~

Comandos no chat:
- Digite normalmente para enviar mensagens
- Use quit para sair do fórum

Para administradores (servidor)

Execução:
./server

Mensagem:
Estamos online na porta: 12345
Esperando em poll()

Nova conexão, file descriptor 1
Cliente [Nome] registrado no indice 1
Recebidos [X] bytes
Inserido: [mensagem] no banco de dados
Encerramento

Encerramento:

Pressione Ctrl+C para encerrar o servidor

Funcionalidades Principais

1. Autenticação Integrada
   - Alunos: Matrícula + senha do Pergamum
   - Professores: Nome + email institucional (consulta via API UFV)
   - Cookies de sessão: Gerenciamento automático de sessões

2. Busca de Livros
   - Consulta em tempo real ao catálogo Pergamum
   - Exibição de número de chamada
   - Conversão automática de codificação (ISO-8859-1 para UTF-8)

3. Fóruns por Livro
   - Chat em tempo real para cada livro
   - Histórico de mensagens
   - Identificação por número de chamada

4. Gestão de Perfil
   - Alunos: Curso, admissão, CPF, email, débitos
   - Professores: Órgão, departamento, telefone, e-mail
   - Informações obtidas automaticamente das APIs da UFV

5. Sistema de Débitos (Apenas Alunos – pressionando 2 no menu)
   - Consulta de multas pendentes
   - Formatação em Reais (R$)

Componentes do Sistema:

Estrutura de Arquivos

```
/
├── Makefile              (Script de compilação)
├── server.cpp/.h         (Servidor principal)
├── client.cpp/.h         (Cliente principal)
├── usuario.cpp/.h        (Classe base Usuario)
├── aluno.cpp/.h          (Classe Aluno)
├── professor.cpp/.h      (Classe Professor)
├── livro.cpp/.h          (Classe Livro)
├── chat.cpp/.h           (Classe Chat)
├── biblioteca.cpp/.h     (Classe Biblioteca)
├── mensagem/
│   └── database.cpp/.hpp (Banco de dados SQLite)
├── main_server.cpp       (Ponto de entrada do servidor)
├── main_client.cpp       (Ponto de entrada do cliente)
└── README.md             (Documentação)
```

Classes Principais:

Usuario (Classe Base)
- Gerencia autenticação e dados básicos
- Métodos para buscar livros e obter cookie de sessão

Aluno (Herda de Usuario)
- Adiciona: curso, admissão, sexo, semestre, CPF
- Método searchDebito() para consultar multas

Professor (Herda de Usuario)
- Adiciona: órgão, departamento, telefone
- Autenticação via nome e e-mail institucional

Livro
- Armazena: nome e ID (número de chamada)
- Objeto básico para representação de livros

Chat
- Gerencia participantes de um fórum
- Lista de pares (ID socket, Usuario)

Biblioteca
- “Contêiner” para livros, usuários e chats
- Métodos de busca e adição

Server
- Multiplexação com poll()
- Persistência de mensagens em SQLite

Client
- Interface de linha de comando
- Threads separadas para envio/recebimento

Banco de Dados:

Localização

O banco SQLite é criado automaticamente pelo código em mensagem/database.cpp.

Operações
- **Inserção**: Ao receber mensagem no chat
- **Consulta**: Ao entrar no fórum (histórico)
- **Escape SQL**: Função escapeSql() previne injeção

⚠ ***Tratamento de Erros:***

Exemplo de exceção lançada:

```
class Saiu_do_chat : public std::runtime_error
// Lançada quando usuário desconecta
```

Recuperação de Erros

1. **Erro de conexão**: Reconexão automática não implementada

2. **Erro de autenticação**: Retorna ao prompt de login
3. **Erro de servidor**: Cliente é encerrado
4. **Timeout**: Servidor usa timeout de 5 minutos no poll()

Logs

- Console do servidor mostra todas as operações
- Erros são escritos em std::cerr

Notas de Desenvolvimento (dev notes):

Dependências Externas

- **cURL**: Requisições HTTP para Pergamum e APIs UFV
- **SQLite3**: Armazenamento persistente de mensagens
- **nlohmann/json**: Parsing de respostas JSON

Codificação

- **C++**: Utiliza features modernas
- **Polimorfismo**: Usuario como classe base
- **RAII**: Gerenciamento automático de recursos
- **Multithreading**: Comunicação assíncrona cliente/servidor

Licença e Atribuições

O sistema foi baseado no Pergamum UFV (APIs públicas da Universidade Federal de Viçosa). Desenvolvido como projeto acadêmico

Uso Acadêmico

Este software é destinado para fins educacionais.
Distribuído sob a licença MIT. Sinta-se livre para utilizar e contribuir para o projeto.