Escola Superior
de Tecnologia
e Gestão
Politécnico de Coimbra

Relatório de Projeto

Programação Aplicada

Avaliação Periódica

Autor(es):

Guilherme Gonçalves a2022156457@alunos.estgoh.ipc.pt
Hugo Pais a2022129956@alunos.estgoh.ipc.pt

Data: Maio de 2024

Resumo

Este projeto é continuação do anteriormente desenvolvido para implementação de uma aplicação com funcionalidades para gestão de obras literárias. Mantêm-se os princípios de Programação Orientada a Objetos (POO), mas pressupõe-se que a comunicação seja em rede. A aplicação é composta por um nó cliente e um nó servidor, onde o cliente realiza operações de autenticação, inserção, consulta de dados de obras literárias e pedidos de revisão, comunicando com o servidor através de sockets. O servidor armazena todas as informações numa base de dados relacional e responde às solicitações do cliente conforme o protocolo de comunicação definido. A interface de utilizador é em modo texto (i.e., consola), proporcionando uma interação eficiente e intuitiva. A nível geral, o projeto integra armazenamento persistente, comunicação em rede e gestão de dados, aplicando conceitos avançados de POO e garantindo uma solução eficiente e escalável. Os resultados destacam a eficaz gestão de autores, das suas obras, processos de revisão e a nível de conclusão ressalta-se a robustez e pertinência do projeto desenvolvido. O trabalho futuro irá incidir sobre melhorias a nível da interação com o utilizador, nomeadamente na utilização de um ambiente gráfico.

Palavras-chave

Aplicação, Autores, Obras Literárias, Processo de Revisão, Revisores, Sockets, Utilizadores.

Índice

Lista de Figuras

Lista de Tabelas

1. Introdução

O presente projeto tem como objetivo expandir as funcionalidades do trabalho anterior, nomeadamente, desenvolvendo uma interface para os autores, que permita a gestão de obras literárias e respetivos processos de revisão através de comunicação em rede (concretamente, por sockets). Esta interface possibilitará a comunicação eficiente entre autores e o servidor, permitindo a submissão de obras para análise, o acompanhamento de processos de revisão, a edição de dados pessoais e a pesquisa / listagem de obras literárias.

A aplicação continuará a ser desenvolvida em Java, seguindo o paradigma da Programação Orientada a Objetos (POO). Este projeto aproveita os conhecimentos adquiridos na Unidade Curricular de Programação Aplicada, do curso de licenciatura em Engenharia Informática na Escola Superior de Tecnologia e Gestão de Oliveira do Hospital (ESTGOH), Unidade Orgânica do Instituto Politécnico de Coimbra. Entre os conceitos aplicados estão objetos e classes, tratamento de exceções, herança e polimorfismo, manipulação de ficheiros para armazenamento persistente dos dados, manipulação de bases de dados relacionais via Java Database Connectivity (JDBC) e a comunicação por sockets.

O foco deste trabalho é a implementação da interface de autores, com comunicação em rede através de sockets. O cliente (interface do autor) não armazenará informações localmente, mas comunicará com o servidor para operações como autenticação, inserção, consulta de obras literárias e pedidos de revisão. A comunicação seguirá um protocolo definido, assegurando que cliente e servidor compreendam e respondam corretamente às solicitações.

A interface para interação com o utilizador é a parte central da aplicação, onde todo o menu e suas funcionalidades estão dispostos em modo texto e são facilmente acedidos. O projeto foi planeado, estruturado, concebido, testado e distribuído considerando os requisitos iniciais, os tempos estabelecidos e a pertinência, oportunidade e necessidade que existe no mercado.

Nos próximos capítulos serão apresentadas mais informações sobre o desenvolvimento e as características do projeto.

2. Estado da Arte

O desenvolvimento de aplicações voltadas para a gestão de documentos é amplamente explorado por diversos softwares. Todavia, soluções específicas desenhadas à medida, para obras literárias, que permitam uma gestão completa desde a submissão até a publicação, sem incluir funcionalidades desnecessárias, são escassas no mercado.

Neste sentido, a abordagem implementada tentou ir ao encontro das necessidades do cliente (i.e., uma Editora Portuguesa conceituada) que, preenchendo uma lacuna no mercado e oferecendo uma solução integrada, permite a submissão de obras, apresenta um acompanhamento do processo e viabiliza a coordenação e aprovação das revisões em curso.

Adicionalmente, nesta fase do projeto, a aplicação desenvolvida diferencia-se da anterior por proporcionar uma interface para autores com comunicação por sockets para gestão de obras literárias de forma integrada e direta com o servidor.

A comunicação cliente-servidor por sockets e a utilização de um protocolo de comunicação específico garantem a segurança e a eficiência na troca de informações, tornando a aplicação versátil e adequada para uso em situações de acesso remoto. A utilização de Java, integrando Programação Orientada a Objetos (POO), garante a escalabilidade e a manutenção da aplicação, facilitando futuras melhorias e adaptações conforme as necessidades da editora.

Deste modo, a abordagem adotada neste projeto é inovadora e relevante, preenchendo uma lacuna existente no mercado e apresentando uma solução por medida eficiente que atende tanto às necessidades dos autores quanto das editoras.

3. Objetivos e Metodologias

Os objetivos consignados ao projeto balizam a metodologia e ações desenvolvidas e, indiretamente, estruturam e planificam a conceção, codificação e implementação do trabalho. A execução desta fase foi alicerçada no briefing dado pelo cliente e nos requisitos por ele considerados, concretamente:

DESCRIÇÃO GLOBAL DA APLICAÇÃO

Ações

[R01] Deve ser disponibilizada uma aplicação cliente que permita realizar as ações associadas ao utilizador autor, nomeadamente:

- autenticar-se, embora não necessite de implementar o registo;
- consultar e alterar informação pessoal;
- inserir informação de uma obra;
- pesquisar e consultar informação de uma obra;
- pesquisar e consultar pedido de uma revisão;
- listar as obras associadas ao autor;
- listar os pedidos de revisão.

[R02] A aplicação cliente não deve armazenar localmente qualquer informação.

A aplicação cliente deve comunicar através da rede com a aplicação servidor para manipular qualquer informação, armazenada na base de dados. A aplicação cliente deve implementar o nó cliente, enquanto a aplicação servidor deve implementar o nó servidor (descritos na secção "Comunicação em rede"). A aplicação servidor corresponde à aplicação desenvolvida no projeto anterior e que deve ser adaptada, de forma a comunicar através da rede.

[R03] Não é necessário garantir que a aplicação servidor realize outras funções no momento em que comunica com aplicação cliente. Ou seja, não é necessário um ambiente multithreading.

[R04] A aplicação servidor deve informar o utilizador dos pedidos que recebe através da rede e da informação que envia à aplicação cliente.

COMUNICAÇÃO EM REDE

Descrição Geral

[R05] No caso de nó servidor, a aplicação fica a aguardar o pedido de ligação de uma aplicação cliente (nó cliente). No caso de nó cliente, a aplicação realiza um pedido de ligação ao nó servidor.

[R06] Na opção de nó servidor deve ser possível indicar o porto para receber pedidos de ligação.

[R07] Na opção de nó cliente deve ser possível indicar o IP e porto para remeter o pedido de ligação.

[R08] Tanto no nó servidor como no nó cliente a aplicação deve indicar o IP local da respetiva máquina.

[R09] Quando um servidor aceita um pedido de ligação de um cliente, tanto o servidor como o cliente devem apresentar essa informação na interface.

[R10] O nó servidor armazena toda a informação na base de dados. Por seu lado, o nó cliente não armazena qualquer informação.

[R11] Após o estabelecimento de ligação todas as mensagens remetidas pelo nó servidor e pelo nó cliente devem ser iniciadas com a expressão <login>, onde login representa o cliente ou servidor. Desta forma, tanto o nó cliente como o nó servidor sabem qual é o interlocutor remoto.

[R12] Todas as mensagens terminam com ponto-e-vírgula (;). Todas as mensagens devem incluir os caracteres < e >, bem como os espaços indicados.

Sequência de comunicação

[R13] A primeira mensagem logo após o estabelecimento de comunicação deverá ser iniciada pelo servidor com a sintaxe <login> <hello>;

O cliente responderá da mesma forma.

Se o servidor não receber como resposta uma mensagem do tipo <hello>, deve remeter novamente a sua mensagem <hello> até receber a resposta correta.

Se o servidor receber a resposta correta do cliente deve enviar o comando <login> <ack>;

[R14] A comunicação termina quando o nó cliente envia a mensagem <login> <bye>;

[R15] O nó cliente envia pedidos ao nó servidor com a seguinte estrutura: <login> <acção> [<argumentos>]; descritos nos pontos seguintes. O nó servidor responde com um comando semelhante, disponibilizando a informação solicitada.
[R16] Para autenticar, o nó cliente envia o comando <login> <autenticar> <username,password>;
Caso a autenticação seja realizada com sucesso, o nó servidor responde com <login> <autenticar> <success>; Caso contrário, o nó servidor responde com <login> <autenticar> <fail>;
[R17] Para consultar os dados pessoais, o nó cliente envia o comando <login> <info>; O nó servidor responde com <login> <info> <username,password,nome,email,estado,nif,telefone,morada>;
[R18] Para alterar os dados pessoais, o nó cliente envia o comando <login> <update> <username,password,nome,email,estado,nif,telefone,morada>; O nó servidor responde com: <login> <update> <ok>;
[R19] Para inserir uma nova obra, o nó cliente envia o comando <login> <inserir> <obra> <titulo,estilo_literario,tipo,num_paginas,num_palavras,ISBN,num_edicao>; O autor da obra deve ser o utilizador autenticado e a data de submissão a data do sistema. O nó servidor responde com: <login> <inserir> <obra> <ok>;
[R20] Para pesquisar uma obra, o nó cliente envia o comando <login> <pesquisa> <obra> <titulo>; O nó servidor responde com <login> <pesquisa> <obra> <titulo,estilo_literario,tipo,num_paginas, num_palavras,ISBN,num_edicao,data_submissao>; Caso não exista a obra o nó servidor responde com <login> <pesquisa> <obra> <fail>;
[R21] Para pesquisar uma revisão, o nó cliente envia o comando <login> <pesquisa> <revisao> <num_serie>; O nó servidor responde com <login> <pesquisa> <revisao> <gestor,revisor,data_ realizacao,tempo_decorrido,observacoes,custo,estado>; Caso não exista a revisão o nó servidor responde com <login> <pesquisa> <revisão> <fail>;
[R22] Para listar todas as obras do autor, o nó cliente envia o comando <login> <listar> <obra>; O nó servidor responde com <login> <listar> <obra> <obra1,obra2,…,obraN> ; Onde obraX corresponde à informação de cada obra associada ao autor, sendo uma listagem. Caso não exista a obra o nó servidor responde com <login> <listar> <obra> <fail>;
[R23] Para listar todas as revisões associadas ao autor, o nó cliente envia o comando <login> <listar> <revisao>; O nó servidor responde com <login> <listar> <revisao> <revisao1, revisao2,…, revisaoN> ; Onde revisaoM corresponde à informação de cada revisão associada a obras do autor, sendo uma listagem. Caso não exista a obra o nó servidor responde com <login> <listar> <revisao> <fail>;
[R24] Sempre que o nó cliente recebe informação do nó servidor, deve responder com o comando <login> <ack>; Por sua vez, o nó servidor deve responder <login> <ack>;

GESTÃO GERAL DA APLICAÇÃO
Armazenamento persistente de dados
[R25] Devem ser implementados métodos de armazenamento persistentes dos dados do lado do servidor, desenvolvidos no projeto anterior. Nenhuma informação deve ser armazenada do lado cliente.

Interação com o utilizador
[R26] Disponibilizar uma interface em modo texto onde o utilizador possa interagir e controlar a aplicação. A organização da interface e escolha de componentes fica ao critério do aluno. Soluções eficientes de menus, respeitando as regras de interação com o utilizador serão bonificadas.

O objetivo central visa a expansão de funcionalidades da aplicação anteriormente desenvolvida, nomeadamente a comunicação entre autores e servidor por sockets.

A metodologia implementada para esta fase de desenvolvimento teve várias etapas de execução. A primeira visou a construção do menu associado à interface autores. A segunda a configuração do nó relativo à comunicação na parte do servidor, verificando-se quais dos métodos desenvolvidos teriam de ser chamados para resposta às necessidades do utilizador. A terceira e última, a integração do trabalho realizado no que já estava implementado, procedendo-se à testagem das funcionalidades e às

correções necessárias ao bom funcionamento do sistema. Neste projeto foram alocados dois desenvolvedores e o trabalho foi igualmente repartido entre eles.

3.1. Ferramentas e Tecnologias

A constituição deste projeto recorreu ao uso de diversas plataformas para dar resposta às necessidades de desenvolvimento. O Eclipse foi o Ambiente de Desenvolvimento Integrado (IDE) selecionado. A opção cingiu-se ao facto de ser aplicação de código aberto que oferece uma ampla gama de recursos, incluindo suporte a várias linguagens de programação, como Java. Além disso a flexibilidade e extensibilidade tornam-no numa escolha sólida para a configuração deste projeto.

Importa salientar que para agilizar o progresso e facilitar a colaboração, foi utilizada a plataforma online Replit. O Replit oferece um ambiente de desenvolvimento online com um IDE integrado que permite aos desenvolvedores a programação simultânea diretamente no navegador. Esta plataforma dispõe de recursos de colaboração em tempo real, tornando-a especialmente útil na implementação de softwares construídos em equipa.

Todos os softwares utilizados foram selecionados com base nas necessidades do projeto e por se considerarem ferramentas robustas, eficientes e colaborativas que atendem às exigências e requisitos considerados.

3.2. Planeamento

O planeamento foi alicerçado nas etapas definidas e apresentadas na metodologia com objetivo de suprir os requisitos e as funcionalidades estabelecidas. A calendarização inicialmente prevista não se mostrou díspar face à execução propriamente dita.

Tabela 1 - Planeamento Previsto

	01/05	08/05	15/05	22/05	26/05	Previsão Tempo
1ª Etapa						2 horas
2ª Etapa						25 horas
3ª Etapa						5 horas
Relatório						4 horas

A realização do projeto não se mostrou complexa, não obstante foi morosa dados os outros trabalhos em andamento pela equipa de desenvolvimento. Não obstante, não se

verificou nenhum comprometimento a nível da execução uma vez que todos os requisitos foram implementados.

A divisão do trabalho entre os desenvolvedores foi devidamente articulada e considerou plataformas colaborativas online para que o processo fosse mais célere e cada interveniente pudesse apoiar, intervir, contactar, ou analisar o código conjeturado pelo outro desenvolvedor.

Em suma o planeamento no global cumpriu-se, foi eficaz e mostrou-se adequado ao projeto.

4. Trabalho Desenvolvido

Neste capítulo será apresentada uma visão ao detalhe daquilo que foi desenvolvido no projeto, descrevendo não só o que foi realizado em termos de classes e métodos, como o pensamento para a sua implementação.

4.1. Requisitos Implementados

Os requisitos apresentados no capítulo objetivos e metodologias foram integralmente implementados, mostrando-se uma consonância entre solicitações do cliente e implementações na plataforma.

4.2. Classes e *Packages*

O projeto foi estruturado com base em vários ficheiros sendo que, cada um, contem uma classe relativa a uma determinada funcionalidade. O título é francamente sugestivo o que confere uma maior perceção sobre a forma como o código se organiza e sobre o local onde as estruturas são manipuladas. A maioria das classes presentes no atual projeto já se encontravam implementadas e descritas no trabalho anterior (i.e., classe "Main.java", "Utilizadores.java", "Revisores.java", "Autores.java", "Revisao.java", "Obras.java", "Anotacoes.java", "Logs.java", "LigacaoBD.java", "GereUtilizadores.java", "GereRevisoes.java", "GereObras.java", "GereNotificacoes.java", "GereLogs.java") pelo que será dado enfoque às novas classes aplicadas.

Classe "Client.java"

A classe Client.java é uma classe relevante nesta fase do projeto. É responsável pela execução do programa na interface cliente para este comunicar com o servidor aquando da gestão das suas obras literárias. Esta classe contém o método main, que inicializa a conexão com o servidor, gere a comunicação e apresenta um menu de opções disponíveis aos autores. A seguir, são detalhados os principais componentes e funcionalidades desta classe.

Inicialização e Conexão ao Servidor:
O método main começa por solicitar ao utilizador o endereço IP e o porto associados ao servidor. Uma vez definidos estes parâmetros o cliente tenta estabelecer a conexão com o servidor utilizando um objeto Socket.

Configuração dos Streams de Entrada e Saída:

Após a conexão, são configurados os fluxos de entrada (BufferedReader) e saída (PrintWriter) para comunicação com o servidor.

Comunicação Inicial:
O cliente envia e recebe mensagens iniciais para estabelecer uma conexão bem-sucedida. Por exemplo, a mensagem "<hello>" é usada para confirmar a comunicação entre cliente e servidor.

Interação com o Utilizador:
O nó cliente apresenta um menu aos autores, permitindo que escolham diferentes opções, como autenticação, inserção de obras, pesquisa, entre outras. Dependendo das escolhas, a função correspondente é chamada para enviar a solicitação apropriada ao servidor.
Cada opção do menu é tratada por métodos específicos que formatam a mensagem a ser enviada ao servidor. Por exemplo, opcao1() procede à autenticação do usuário, enquanto a opcao3() permite a inserção de uma nova obra.

Encerramento:
O cliente pode encerrar a conexão enviando a mensagem de saída apropriada e fechando todos os recursos de rede utilizados.

Classe "Server.java"

A classe Server implementa o nó servidor que aceita a ligação com clientes e processa as várias operações que vão sendo solicitadas pelos autores. Vão desde a autenticação de utilizadores, à manipulação de dados pessoais e à gestão de obras literárias e respetivas revisões.

Métodos

servidor() – Inicia o servidor e gere a comunicação com o cliente. Aceita conexões, recebe e envia mensagens, e processa comandos específicos.

autenticacao() – Realiza a autenticação do utilizador / autor.

consultaDadosPessoais() – Consulta e apresenta os dados do autor autenticado.

editaDadosPessoais() – Atualiza os dados pessoais do autor autenticado.

insereObra() – Adiciona uma nova obra do autor no sistema

pesquisaObra() – Pesquisa obras do autor com base no título.

pesquisaRevisao() – Pesquisa pedidos de revisão do autor com base no número de série.

listarObras() – Lista todas as obras do autor autenticado.

listarRevisoes() – Lista todos os pedidos de revisão do autor autenticado.

funcaoSplit() – Divide em parâmetros a mensagem recebida.

lerDadosInt() – Lê um número inteiro da consola.

4.3. Algoritmos

A realização deste projeto, foi alicerçada maioritariamente nos algoritmos anteriormente desenvolvidos consolidando o correto funcionamento e tendo em conta que nas testagens as diversas operações de gestão de obras literárias estavam operacionais. De seguida representam-se alguns dos algoritmos implementados no trabalho:

- O algoritmo de registo e de autenticação de forma a poder validar se determinados atributos estão disponíveis para se utilizar e se as credenciais introduzidas pelo utilizador na hora da autenticação estão corretas.
- A aprovação de registos, no qual o gestor pode aprovar os registos de novos utilizadores, validando neste caso as informações deste.
- A possibilidade de ativar ou desativar um utilizador, a pedido de este ou devido a outros fatores.
- A gestão de revisões, no qual é necessário a criação, a aprovação e a execução de revisões, que envolvem todos os utilizadores da aplicação, de forma a garantir uma maior eficiência nos seus processos.
- A gestão das obras, para poder adicionar, atualizar e listá-las.
- A gestão dos utilizadores, de forma a poder adicionar, atualizar e listar os utilizadores.
- O algoritmo de gestão de notificações para os utilizadores poderem visualizar as suas notificações referentes a alguns passos do processo de revisão entre outras.
- E o algoritmo de gestão de logs de forma a ser possível a visualização de um certo utilizador em casos de necessidade.
- Para guardar os dados necessários ao longo dos processos, um dos processos mais importantes, recorreu-se a uma base de dados que garantem que as informações dos utilizadores, das revisões, das obras, das notificações e dos logs sejam armazenados corretamente e recuperados também de forma certa.

Com este tipo de algoritmos mantém-se um correto funcionamento do programa, de forma a que este funcione de forma acertada de forma geral.

4.4. Armazenamento de Dados

Para este projeto de forma a garantir a persistência dos dados, foi utilizada a base de dados anteriormente criada, mantendo as tabelas e atributos considerados como

necessários para a caracterização dos objetos. Também se utilizou um ficheiro de texto, de propriedades, de forma a armazenar os dados de acesso à base de dados.

BaseDados.properties: Este ficheiro guarda os dados para que se consiga aceder à base de dados de forma a inserir, visualizar ou alterar os dados dependendo daquilo que se pretenda.

Base de Dados: base de dados onde é armazenada toda a informação necessária para o programa.

A construção da base de dados teve por princípio o diagrama Entidade Relacionamento, onde se estruturou as informações relevantes do sistema e a forma como estas se relacionam. As tabelas a implementar na base de dados, as variáveis e suas especificidades (e.g., se é primary key, chave forasteira) já foram esquematizadas e sintetizadas anteriormente pelos diagramas concetual e físico, porém voltam a ser expostos em seguida.

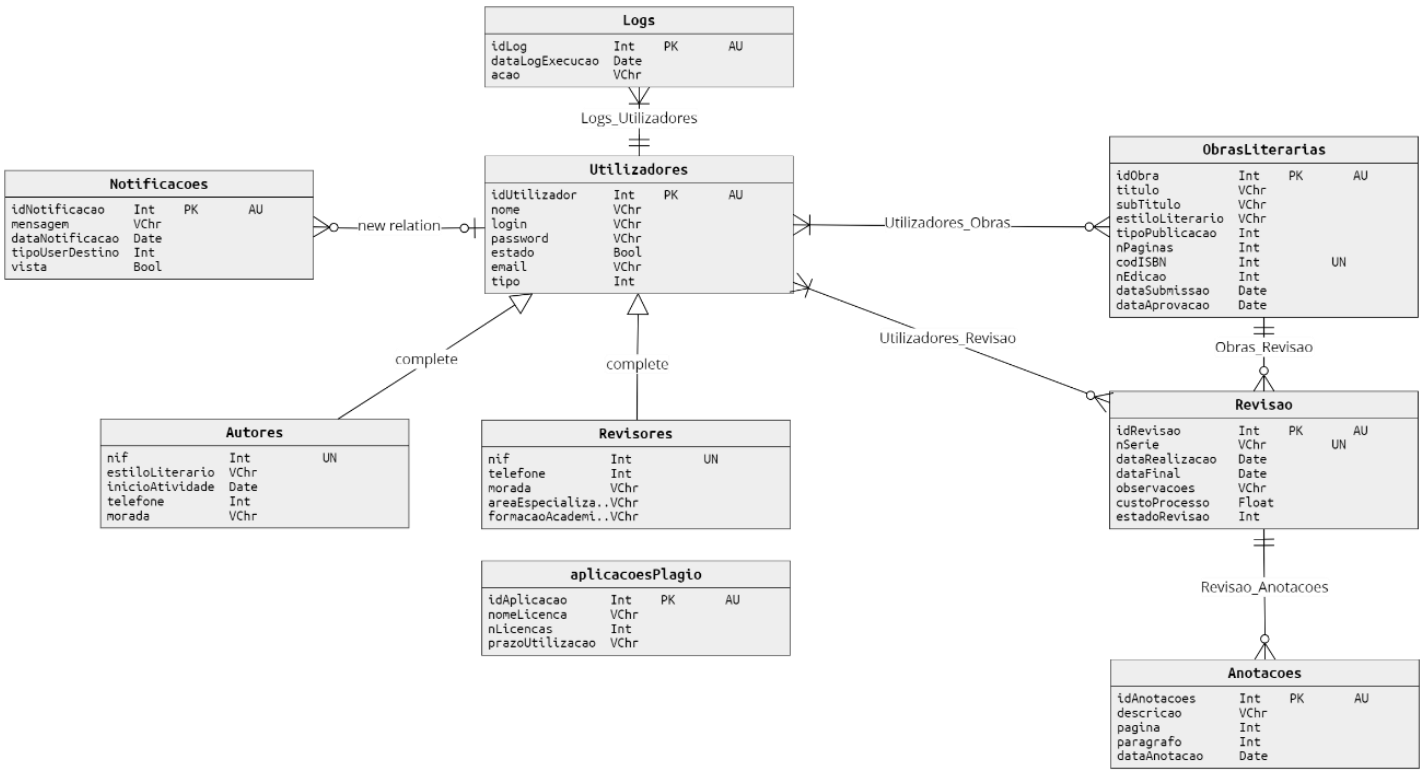

Figura 4-1 – ***Diagrama Conceptual.***

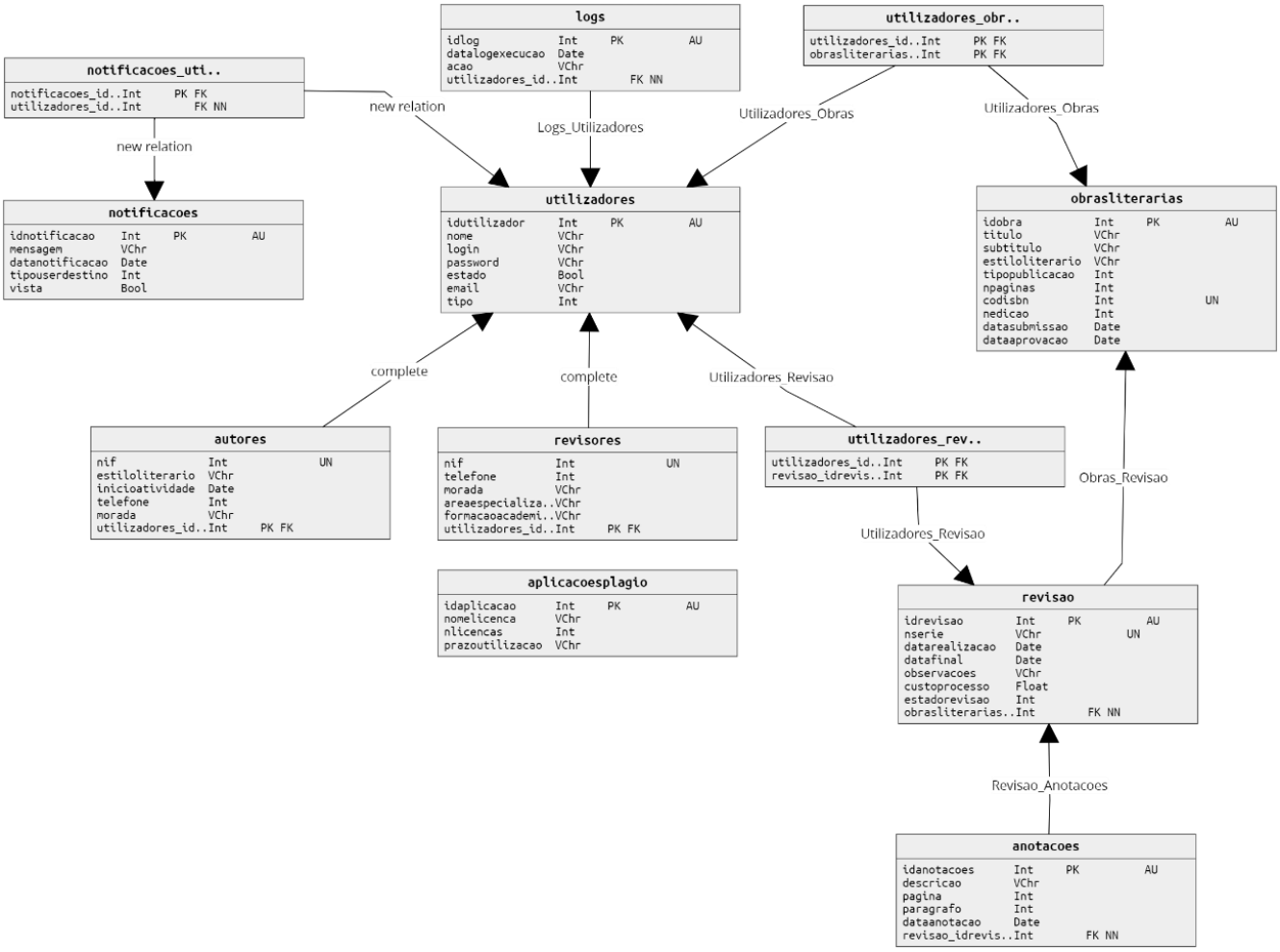

Figura 4-2 – ***Diagrama Físico.***

4.5. Procedimentos de Teste

Os procedimentos implementados e os testes realizados serviram para garantir o correto funcionamento da aplicação e também uma boa qualidade e otimização do código em grande parte do programa.

No âmbito deste projeto, foram realizados testes unitários (i.e., relativos a cada opção do menu cliente) para análise das funcionalidades, validação do funcionamento, conexão com o servidor e articulação com os métodos já implementados. Também foram realizados testes para garantir que a interação com o utilizador fosse acessível e entre o nó cliente e servidor fosse conforme o enunciado. Os testes feitos não mostraram erros e denotaram o funcionamento conforme previsto da aplicação.

Estes procedimentos foram considerados de forma sistemática ao longo do desenvolvimento, contribuindo para a robustez e confiabilidade do código.

5. Conclusões

Neste segundo projeto, o objetivo foi aprimorar a aplicação anteriormente desenvolvida, para gestão de processos de revisão de obras literárias, porém com foco na criação de uma interface cliente (destinada aos autores) que se conecta ao servidor via sockets, permitindo a execução de diversas funcionalidades. A implementação implicou o uso de conceitos de programação orientada a objetos, manipulação de bases de dados relacionais utilizando JDBC, e o desenvolvimento de uma comunicação eficiente entre cliente e servidor.

O sistema foi projetado para facilitar a gestão por parte dos autores. A interface cliente foi desenvolvida para lhes permitir interagir com o servidor, submetendo pedidos e recebendo respostas em tempo real, através de uma comunicação baseada em sockets.

O projeto atingiu os objetivos inicialmente propostos, implementando as funcionalidades solicitadas e garantindo uma comunicação estável entre o cliente e o servidor. Os utilizadores podem autenticar-se, consultar e editar dados pessoais, inserir novas obras, pesquisar e listar obras e revisões, tudo através de uma interface cliente que interage diretamente com o servidor.

Este projeto proporcionou uma excelente oportunidade para aplicar e aprofundar conhecimentos adquiridos nas disciplinas de Programação Aplicada e Bases de Dados, bem como desenvolver habilidades práticas em rede e comunicação de dados. A colaboração em grupo e a divisão de tarefas foram essenciais para o sucesso do projeto, refletindo-se na qualidade do sistema desenvolvido.

A tabela a seguir representa o tempo investido por cada membro do grupo no projeto e na disciplina como um todo, oferecendo uma visão clara do envolvimento de cada um na realização do mesmo.

Tabela 2 - Tempo Utilizado com a Unidade Curricular de Programação por Elemento do Grupo.

Aulas	Projeto	Relatório
5h / semana	± 30h	5h

5.1. Forças

As forças deste projeto que as distingue de outras abordagens é o facto da aplicação oferecer uma gestão robusta e eficiente no que toca aos utilizadores, permitindo o acesso a determinadas funcionalidades por comunicação em rede.

5.2. Limitações

Apesar de existirem forças em relação ao trabalho, também existem algumas limitações. A principal limitação encontrada ao longo do trabalho foi por exemplo, devido aos outros projetos em curso que limitaram o tempo disponível não sendo possível ir ao detalhe.

5.3. Trabalho Futuro

Para trabalho futuro, é recomendável de forma a melhorar ainda mais o projeto, implementar por exemplo uma interface gráfica para proporcionar uma experiência visualmente mais apelativa ao utilizador e considerar otimizar o código de forma a melhorar o desempenho da aplicação.

6. Referências

Stackoverflow (2024, março, 18). *Regular Expression for Mobile*. Disponível em: https://stackoverflow.com/questions/45704925/regular-expression-for-mobile-and-starting-with-09-or-04

Replit (2024, janeiro, 08). *How do you clear terminal in Java?*l Disponível em: https://replit.com/talk/ask/How-do-you-clear-terminal-in-Java/46341

Veloso, M. (2023). *Slides e Materiais de Apoio às Aulas*. [PDF de apoio à UC de Base de Dados, lecionada na ESTGOH - IPC].

Veloso, M. (2024). *Slides e Materiais de Apoio às Aulas*. [PDF de apoio à UC de Programação Aplicada, lecionada na ESTGOH - IPC].

Velykozhon, D. (2024, março, 18). *Simple Email Validation in Java*. Disponível em: https://mailtrap.io/blog/java-email-validation/