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: Junho de 2024

Resumo

Este projeto pressupõe o desenvolvimento de uma aplicação para gestão de obras literárias em Java, utilizando os princípios da Programação Orientada a Objetos (POO). O objetivo principal é a construção de uma solução abrangente que integre dados inerentes a obras literárias, autores, revisores e respetivos processos de revisão. Para o desenvolvimento foram implementadas classes que permitiram representar os diferentes elementos da aplicação (e.g., obras, autores) e os processos de revisão (i.e., revisão). A gestão e articulação entre os objetos definidos levou à constituição de classes para gestão (e.g., gereUtilizadores, gereRevisoes) que, contendo diferentes métodos para concretização de ações (e.g., pesquisa, listagem, verificação), permitiram o eficaz cumprimento dos objetivos. A manipulação de bases de dados relacionais através de Java Database Connectivity (JDBC), o uso de ficheiros para armazenamento de forma persistente (e.g., das propriedades para ligação à Base de Dados) e a criação da Interface Gráfica para Interação com o Utilizador foram alguns dos requisitos da aplicação. Os resultados destacam a eficaz gestão de autores, obras e processos de revisão e a nível de conclusão ressalta-se a robustez e pertinência do projeto desenvolvido. O trabalho futuro irá incidir sobre atualizações e melhorias contínuas ao trabalho aqui explanado.

Palavras-chave

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

Índice

Lista de Figuras

Lista de Tabelas

1. Introdução

O presente projeto pressupõe melhorias ao desenvolvimento da aplicação anteriormente considerada no trabalho um para gestão de obras literárias e respetivos processos de revisão. A operacionalização já concretizada possibilitava a comunicação entre autores, gestores e revisores, nomeadamente, para gestão de processos de revisão, com possibilidade de submissão de obras, acompanhamento de processos de revisão, pagamento, e respetiva publicação. A aplicação configurada recorria a uma interface em modo texto pouco prática e, esteticamente, pouco apelativa.

O atual projeto mantem algumas premissas anteriores e recorre à linguagem Java, seguindo o paradigma de Programação Orientada a Objetos (POO). A implementação volta a abranger conceitos importantes abordados na Unidade Curricular de Programação Aplicada, na licenciatura em Engenharia Informática da Escola Superior de Tecnologia e Gestão de Oliveira do Hospital (ESTGOH), Unidade Orgânica, do Instituto Politécnico de Coimbra. Concretamente, estão integrados neste projeto competências relativas a objetos e classes, tratamento de exceções, heranças e polimorfismo, manipulação de ficheiros para armazenamento persistente dos dados, manipulação de bases de dados relacionais através de Java Database Connectivity (JDBC), e construção de uma interface gráfica para interação com o utilizador (Veloso, 2024).

A interface gráfica é a componente central da aplicação e deste projeto, permitindo uma interação intuitiva e eficiente entre os utilizadores e o sistema. As melhorias conjeturadas serão planeadas, estruturadas, desenvolvidas, testadas e distribuídas considerando os requisitos estabelecidos, prazos definidos, e as necessidades do mercado.

Os objetivos incluem garantir a aplicação prática dos conhecimentos de programação em JAVA, bem como o desenvolvimento de uma solução funcional e eficaz, revestida por uma interface gráfica simples e intuitiva capaz de viabilizar a correta gestão de obras literárias e dos processos de revisão.

Nos próximos capítulos serão dadas mais informações relativamente ao projeto desenvolvido e suas características.

2. Estado da Arte

O desenvolvimento de aplicações direcionadas à gestão de revisões de documentos encontra-se explorado em diferentes softwares. Todavia, versões desenhadas à medida, que sejam específicas para obras literárias e permitam a gestão completa das obras, do respetivo processo de revisão e não apresentem funcionalidades desnecessárias é algo que não se encontrou no mercado.

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

A incorporação de funcionalidades como listagem de obras, pesquisa de autores, notificações relativas a processos, interação entre revisores e autores, proporcionou uma comunicação efetiva entre os diferentes agentes do serviço. Paralelamente, a interface gráfica desenvolvida dá o aporte necessário para alavancar a utilização do sistema de forma simples e intuitiva. A colocação dos dados numa base de dados e potencial articulação online conferem versatilidade que permitirá a sua utilização do software em qualquer local que ofereça acesso remoto.

A abordagem atual e as melhorias apresentadas permanecem necessárias e relevante dada a ausência de soluções minimalistas associadas a esta área de negócios, preenchendo uma lacuna no campo da gestão de obras literárias e nos processos de revisão.

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 foi alicerçada no briefing dado pelo cliente e nos requisitos por ele considerados, concretamente:

DESCRIÇÃO GLOBAL DA APLICAÇÃO
[R01] Corrigir e completar a aplicação iniciada no projeto 1, cumprindo os respetivos requisitos.

[R02] Não é necessário apresentar mensagens de boas-vindas no arranque e de encerramento, nem estatísticas de utilização, como data, duração ou número de execuções da aplicação.

Gestão de acesso e utilizadores
[R03] Além dos atributos definidos no projeto 1, os utilizadores possuem uma foto, que pode ser inserida ou alterada em qualquer momento. Caso o utilizador não defina a sua foto, deve existir uma imagem por defeito/genérica no sistema.

[R04] No momento de registo deve ser enviado um email a informar que o registo foi realizado.

Ações
[R05] Em todos os processos deve existir uma caixa para inserir observações.

[R06] Após autenticação, deve surgir uma caixa de diálogo ou uma zona na janela principal em realce a indicar a existência de notificações, caso existam.

[**R07]** Deve ser possível imprimir um extrato das ações de um processo (indicação das datas e dos utilizadores que realizaram ações num processo).

Listagens e pesquisas
[R08] Todas as listagens devem permitir visualizar os resultados por páginas de 10 registos cada ou incluir barras de deslocamento.

GESTÃO GERAL DA APLICAÇÃO
Interação com o utilizador
[R09] Toda a interação com o utilizador (apresentação de informação ou manipulação de dados) deverá ser realizada através de interfaces gráficas (componentes gráficas como janelas, painéis, caixas de diálogos, menus, abas, tabelas, botões, caixas de texto, etc.) disponíveis através da biblioteca gráfica JavaSwing / JavaFX.

A organização da interface e escolha de componentes fica ao critério do aluno, no entanto devem ser intuitivas, limpas e claras. Soluções eficientes de disposição e transição de componentes, respeitando as regras de interacção com o utilizador serão valorizadas.

[R10] Devem ser usadas componentes gráficas adequadas à informação a apresentar. Por exemplo, na apresentação de dados estruturados (listagens) devem ser usadas listas (JList) ou tabelas (JTable) e não em texto simples (JLabel ou JTextField).

[R11] Os diversos formulários devem possuir uma componente de ajuda. Sobre cada elemento da interface (e.g. campo de texto, botões) deve existir uma região onde sobrepondo o rato permita o aparecimento de uma nova janela com a informação de ajuda (tooltips).

O objetivo central impele à melhoria do trabalho já realizado, implementação da interface gráfica, que cumpra os pressupostos da empresa cliente e seja uma mais valia para o seu negócio e seus utilizadores.

A metodologia implementada teve várias etapas de execução. A primeira visou a revisão do Diagrama Entidade Relacionamento, classes, objetos, atributos necessários para adaptar o projeto. A segunda, a construção do script para criação da base de dados e a definição adaptação de alguns métodos para responderem às novas funcionalidades. A terceira, a configuração das interfaces gráficas (e.g., ecrã inicial, dashboard do utilizador) e desenvolvimento de funções, componentes e ações que respondessem aos requisitos indicados pela empresa cliente. A última fase foi para testagens e correções do sistema que permitam a redistribuição. Foram alocados dois desenvolvedores ao projeto 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 e modelagem. 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.

Para agilizar o progresso e facilitar a colaboração, foi utilizada a plataforma online Replit que 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.

Por fim, para reconfiguração dos Diagramas Entidade Relacionamento e obtenção do script de criação da base de dados, foi utilizada a plataforma Onda. Esta plataforma é uma ferramenta dedicada à modelação de dados que permite facilmente criar os diagramas conceptuais e físicos das aplicações em causa dispondo ainda de uma funcionalidade que permite gerar automaticamente o script SQL capaz de criar a base de dados. A sua interface intuitiva e recursos avançados levaram a que fosse considerado neste projeto.

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

	07/06	09/06	11/06	13/06	15/06	17/06	Previsão Tempo
1ª Fase							01 horas
2ª Fase							04 horas
3ª Fase							25 horas
4ª Fase							05 horas
Relatório							05 horas

A realização das melhorias previstas neste projeto dado o pouco tempo para execução converteu-se num enorme desafio. Afinal era necessário aprender a utilizar e implementar novas bibliotecas para desenhar a interface gráfica. Todavia não se verificaram comprometimentos a nível da execução uma vez que todos os requisitos foram implementados, embora um deles não esteja totalmente conforme o estipulado.

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 dada a extensão do 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. Destacar que não foi incluído diagrama de classes neste capítulo, uma vez que esse conteúdo programático só decorreu este semestre na Unidade Curricular de Engenharia de Software e até à data de fecho deste trabalho era impossível integrar conteúdos tão recentes que mereciam uma abordagem mais cuidada.

4.1. Requisitos Implementados

Os requisitos apresentados no capítulo objetivos e metodologias foram todos implementados, todavia o requisito 7 (i.e., imprimir extrato das ações de um processo) foi concretizado com impressão associada a outro método implementado no gereRevisões. No desenvolvimento anterior não se encontravam previstos métodos que dessem resposta ao solicitado e na nossa perspetiva seria útil reformular até a base de dados para o efeito. Nesse sentido, reportou-se o facto à empresa cliente e a concretização do método em pleno fica para uma atualização futura.

4.2. Classes e *Packages*

O projeto foi estruturado com base em vários ficheiros, no qual, cada ficheiro, de forma a estarem mais organizados, continha uma classe referente a uma certa funcionalidade. De seguida estarão representadas todas as classes utilizadas no mesmo, com os principais métodos implementados no trabalho, explicados com base na sua função.

Classe "EcraInicial.java"

Esta classe representa a tela inicial da aplicação, onde os utilizadores podem optar por fazer login ou registar-se. Ela configura a interface gráfica inicial e direciona os utilizadores para os formulários de login ou registo, conforme a sua escolha. É a porta de entrada para o sistema, garantindo que apenas utilizadores autenticados possam aceder às funcionalidades do sistema.

Classe "FormLogin.java"

Esta classe implementa o formulário de login, permitindo aos utilizadores autenticarem-se no sistema. O formulário solicita o login e a password, validando as credenciais fornecidas através da ligação à base de dados. Caso as credenciais sejam válidas, o

utilizador é redirecionado para a Dashboard com as permissões adequadas ao seu tipo de conta.

Classe "FormRegisto.java"

Esta classe implementa o formulário de registo, onde novos utilizadores podem criar contas no sistema. O formulário solicita informações como login (i.e., username), password, nome, email, e tipo de utilizador (autor, revisor ou gestor). Após a submissão do formulário, os dados são validados e armazenados na base de dados, criando um novo utilizador no sistema. Inclui também a funcionalidade de envio de email de confirmação (Atenção: Deve configurar-se no código a palavra associada ao email).

Classe "Dashboard.java"

Esta classe é responsável pela interface gráfica principal da aplicação, onde todas as interações dos utilizadores (autores, gestores e revisores) são centralizadas. Através desta interface, os utilizadores podem aceder às diferentes funcionalidades do sistema, como submissão de obras, gestão de processos de revisão, consultas de notificações, entre outros. A classe implementa diversos painéis e botões, utilizando o CardLayout para alternar entre as diferentes vistas e funcionalidades da aplicação.

Classe "Paginacao.java"

Esta classe implementa a funcionalidade de paginação nas listagens da interface gráfica. Permite dividir grandes volumes de dados em páginas menores, facilitando a navegação e a visualização dos dados pelos utilizadores. Inclui métodos para navegar entre as páginas, atualizar a lista exibida e selecionar itens para ações subsequentes.

Classe "Utilizadores.java"

Esta classe, tal como dito na classe anterior, é a superclasse, visto que as outras classes herdam os atributos desta. Representa um utilizador, e contém atributos relativos a este, incluindo informações como login, password, nome, email, entre outros. Tal como as outras classes, esta também contém, formas de obter os atributos, alterar e uma função "toString".

Classe "Revisores.java"

Esta classe herda os atributos da classe "Utilizadores", sendo a classe filha, associada à superclasse "Utilizadores". Foi necessário recorrer à herança uma vez que nesta classe além dos atributos específicos dos "Gestores", existem cinco atributos adicionais (i.e., nif, morada e telefone, área de especialização e formação académica). Tal como nas

classes anteriores esta também apresenta um construtor, para inicializar o objeto, operações que permitem a leitura e a atualização de certos atributos, e também um método "toString" de forma a utilizar-se no retorno de certas funções, em que é necessário obter uma frase com certos dados do objeto.

Classe "Autores.java"

Esta classe, tal como a classe "Revisores" herda os atributos da classe "Utilizadores". Nela foram considerados cinco atributos adicionais (i.e., nif, morada e telefone, estilo literário e início da atividade). Conforme a classe revisores há um construtor, métodos para ler e atualizar certos atributos e um método "toString" para obter uma frase com certos dados do objeto.

Classe "Revisao.java"

Esta classe modela as revisões no sistema, com as informações inerentes ao pedido de revisão em causa. Tal como as classes anteriores este objeto reúne um conjunto de atributos característicos (e.g., idRevisao, nSerie, dataRealizacao, dataFinal) e integra métodos para obter informações sobre um atributo, fazer-lhes atualizações e o método "toString" com o formato desejado para imprimir mensagens na consola.

Classe "Obras.java"

Nesta classe consideram-se os atributos que caracterizam e integram o objeto relativo às obras literárias (e.g., idObra, titulo, subtítulo, nPaginas, codISBN) e tal como nas classes anteriores existe o construtor e métodos de leitura e atualização dos atributos. É também estabelecida a formatação para o método toString que faz uma representação escrita das informações a serem apresentadas ao utilizador.

Classe "Anotacoes.java"

Nesta classe são representadas as anotações associadas a uma obra literária, nomeadamente as que derivam dos processos de revisão. Na estrutura da classe encontra-se apenas o construtor uma vez que a implementação desta funcionalidade não foi completamente introduzida na interação com revisores e autores.

Classe "Logs.java"

Esta classe é onde se integram os atributos necessários ao armazenamento das informações de log (i.e., ações e data). Este método relacionado com a classe utilizadores irá permitir consolidar informações relevantes sobre a ação dos utilizadores no sistema.

Classe "LigacaoBD.java"

Esta classe integra os métodos responsáveis pela ligação à base de dados. Na sua estrutura encontramos um método para carregar as propriedades da ligação do ficheiro BaseDados.Properties, conectar à base de dados, desconectar, atualizar as propriedades de ligação e reescrevê-las no ficheiro.

Classe "GereUtilizadores"

Nesta classe é onde estão todos os métodos que permitem gerir as várias operações relacionadas aos utilizadores. Alguns dos métodos criados e implementados foram por exemplo:

- "adicionarInfoAutorBD" – método que permite adicionar um novo autor à base de dados;
- "verificaLoginUtilizador" – método implementado no momento da autenticação, que verifica se o utilizador com um determinado login e password inseridos pelo utilizador, existe na base de dados. Se o utilizador existir, é devolvido esse mesmo utilizador que fica com acesso às diferentes opções tendo em conta o seu tipo. Caso não exista, retorna ao menu inicial sem acesso às áreas restritas.
- "listarUtilizadoresPendentes" – método que retorna a listagem com todos os utilizadores pendentes para o gestor aprovar.

Classe "GereRevisoes"

Na classe "GereRevisoes" estão implementados os métodos capazes de gerir as várias operações relacionadas aos pedidos de revisao. Alguns dos métodos criados e implementados foram por exemplo:

- "adicionarPedidoRevisao" – método que permite adicionar um novo pedido de revisão à base de dados para posterior análise pelo gestor;
- "geraNSerie" – método que permite gerar um número de série único para os pedidos de revisão com base na concatenação de um número sequencial com a data do pedido;
- "listarPedidosRevisaoDeAutor" – método que retorna a listagem com todos os pedidos de revisão de um determinado autor.

Classe "GereObras"

Classe responsável por gerir, através dos métodos criados nesta, as várias obras literárias da aplicação. Alguns dos métodos criados e implementados foram por exemplo:

- "pesquisarObrasDeAutorDataSubmissao" – método que permite pesquisar obras de um autor considerando a data de submissão;
- "adicionarObra" – método para inserir uma nova obra literária na base de dados;
- "criaVerificaISBN" – método para criar e verificar a existência de um ISBN na base de dados.

Classe "GereNotificacoes"

Classe responsável pela gestão das notificações do sistema. Alguns dos métodos criados e implementados foram por exemplo:

- "adicionarNotificacao" – método que permite adicionar uma notificação à base de dados;
- "listarNotificacoesGestores" – método para listar as notificações que dizem respeito aos gestores;
- "darVistaNotificacoes" – método para marcar como vista as notificações de um determinado utilizador.

Classe "GereLogs.java"

Classe responsável pelo registo e apresentação das ações que os utilizadores desenvolvem no sistema. Exemplos dos métodos criados são:

- "logAcao" – método que permite adicionar um log à base de dados;
- "listarLogsUtilizador" – método para listar os logs de um determinado utilizador.

4.3. Algoritmos

Na realização deste projeto foram mantidos os algoritmos desenvolvidos anteriormente, tentando que o sistema permaneça a funcionar corretamente e as operações de gestão realizadas sejas fidedignas e seguras. De seguida são apresentados 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 previne-se o funcionamento do programa e tenta garantir-se que os pressupostos e requisitos recolhidos com a empresa cliente.

4.4. Armazenamento de Dados

Para este projeto, tal como no seu antecessor, a garantia para persistência dos dados foi salvaguardada pela utilização de uma base de dados, com várias tabelas e atributos de acordo com as necessidades do projeto e respetivo sistema. 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 estruturaram as informações relevantes do sistema e a forma como as tabelas se relacionam. As tabelas a implementar na base de dados, as variáveis e suas especificidades (e.g., se é primary key, chave forasteira) são esquematizadas e sintetizadas quer no diagrama concetual quer no diagrama físico. Seguidamente apresentam-se os diagramas. A diferença face ao projeto inicial residiu na inclusão do atributo “foto”, na tabela “utilizadores”.

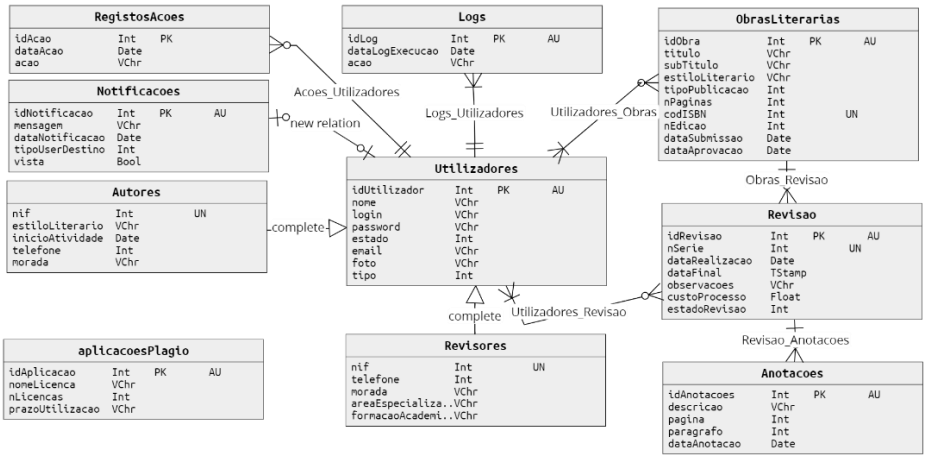

Figura 4-1 – *Diagrama Conceptual.*

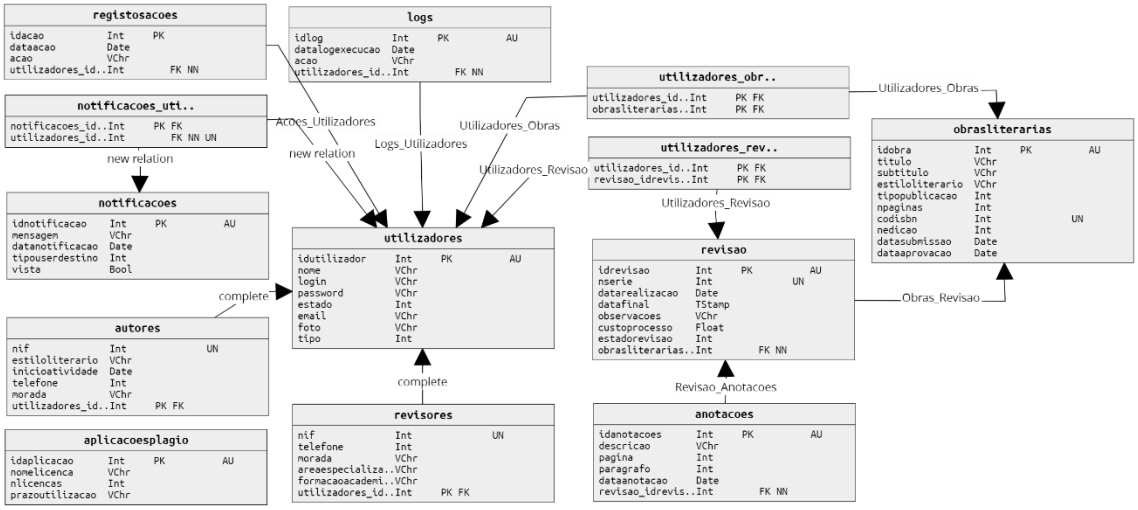

Figura 4-2 – *Diagrama Físico.*

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) para análise das funcionalidades, validação do funcionamento e articulação dos métodos feitos. Também foram realizados testes para garantir que a interação com o utilizador e interface gráfica aplicada eram facilitadoras da ação e demonstravam ser simples e intuitivas. A ideia passava por não criar grande confusão na utilização e mitigar / prevenir erros. Ainda durante os testes, foi verificado se as informações estavam corretas sem qualquer erro entre as diferentes execuções.

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

5. Conclusões

Neste projeto de melhorias, o objetivo foi o aprimoramento da aplicação em linguagem Java dedicada a auxiliar uma editora no processo de revisão de obras literárias pela inserção de uma interface gráfica. Ao longo da implementação desta interface e da configuração de novos requisitos implementados, foi necessário utilizar bases de programação orientada a objetos, manipulação de base de dados relacionais utilizando JDBC e recurso a ficheiro para armazenamento de dados. Além disso, o desenvolvimento de interfaces gráficas foi a parte crucial para o processo de interação com o utilizador. Utilizou-se o Java Swing para a construção das janelas e componentes visuais, competências de gestão de eventos, layout managers e manipulação de elementos gráficos. Bibliotecas adicionais como JavaMail API para envio de emails e funcionalidades de upload de imagens com a utilização de classes como `JFileChooser` e `ImageIcon` também foram integradas, permitindo uma interação mais robusta e amigável para o utilizador final.

A aplicação aborda eficientemente a gestão dos utilizadores, com diferenças entre os três tipos. Uma das coisas que contribuiu para a eficiência do sistema foi a forma coerente de como foram feitas as diversas manipulações e as diferentes Dashboards automaticamente configuradas pelo tipo de utilizador logado.

Podemos concluir, com isto, que o projeto atingiu os objetivos propostos, tirando um ou outro requisito, que podia ter sido implementado de forma mais otimizada, mas que por escassez de tempo não foi possível. A nível geral, o projeto proporcionou uma oportunidade para aplicar e aprimorar os conhecimentos e competências adquiridos ao longo do primeiro semestre nas Unidades Curriculares de Programação e de Bases de Dados.

De seguida estará representada uma tabela com o tempo gasto com este projeto e com a disciplina na globalidade, por elemento do grupo, de forma a ter-se uma noção do quão envolvido foi a realização do mesmo.

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

Aulas	Projeto	Relatório
5h / semana	± 35h	4h

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 registo, a devida autenticação e a sua correta identificação. Outro aspeto importante, foi o facto de colocarmos uma verificação a seguir a cada escolha de opção, para caso o

utilizador se tenha enganado a escolher, ainda tenha a oportunidade de voltar atrás sem ter de obrigatoriamente entrar nessa opção.

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 à falta de tempo para a sua concretização e não conseguirmos ir mais 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 ainda mais completa 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/