Base de Dados

- Relatório do Projeto –

Meta 2

Elementos do Grupo e Contactos :

Ana Inês Mesquita Fidalgo, aimf@student.dei.uc.pt, 2013134819

Alexandra Tomé Leandro, aleandro@student.dei.uc.pt, 2013146088

Manuel Madeira Amado, maba@student.dei.uc.pt, 2006131282

Descrição Breve

Crowdsourcing serve como mecanismo financeiro para apoiar novas ideias, projetos e produtos. Assim, potenciais clientes podem pagar adiantado por produtos e ter acesso a itens promocionais exclusivos enquanto as empresas podem receber dinheiro mais cedo, o que lhes permite avançar com projetos de investimento alto com garantia de terem clientes.

Neste projeto vamos implementamos uma base de dados de uma plataforma de crowdfunding em que os apoiantes podem votar em diferentes alternativas de realização de um projeto consoante o dinheiro que investem no projeto.

A solução encontrada foi a construção de uma api para uma base de dados oracle que é chamada através de uma interface java Swing.

A concorrência é obtida a partir da geração de varias interfaces e estas são as responsáveis pela constante actualização dos dados.

Funcionalidades:

- Criação de Users;

- Login de Users;

- Permitir aos users a criação de projetos a serem apoiados;

    - Cada projeto terá um saldo objetivo a ser atingido num espaço de tempo;

- Possibilidade de users doarem dinheiro a projetos;

    - De acordo com a quantidade de dinheiro doado o user poderá votar numa opção para tomada de decisões;
    - Se o user doar determinado montante objectivo poderá ter direito a Recompensas;

- Cada recompensa tem o seu valor objetivo;

- Envio de mensagens;

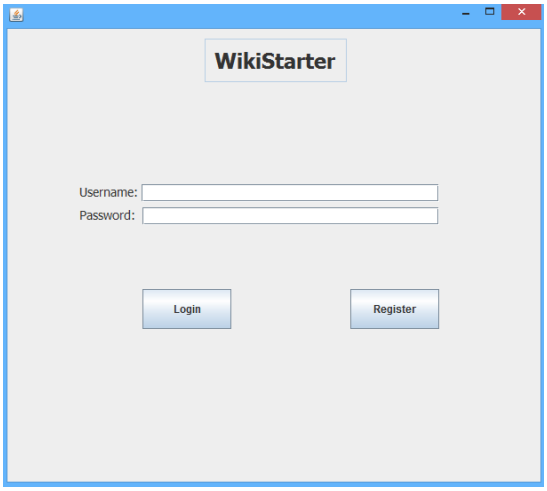
    - Cada projeto tem inbox;

- Continua verificação se projetos atingiram data limite;

- Caso não atinja valor objetivo no prazo todas doações serão devolvidas e tudo ficará sem efeito;

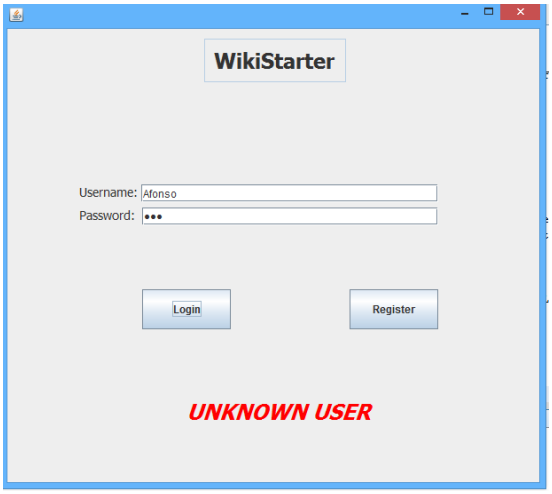
- Caso tenha saldo igual ou superior na data limite então as doações são debitadas e as recompensas entregues.

Manual Utilizador

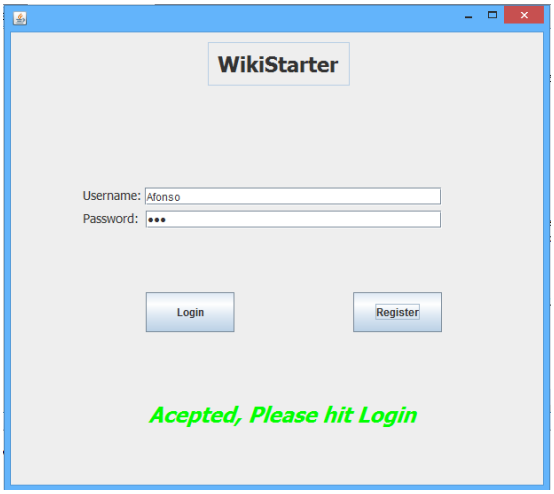


1– O menu Log in onde o utilizador insere o seu username e a sua password, caso não esteja já registado pode faze-lo.

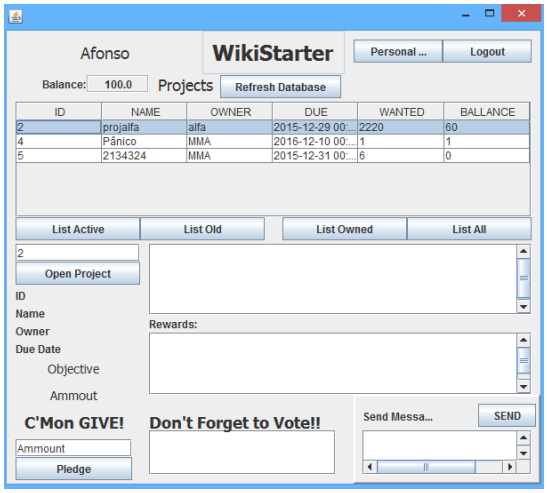
2- Caso o utilizador insira a password mal é apresentada uma mensagem de erro.



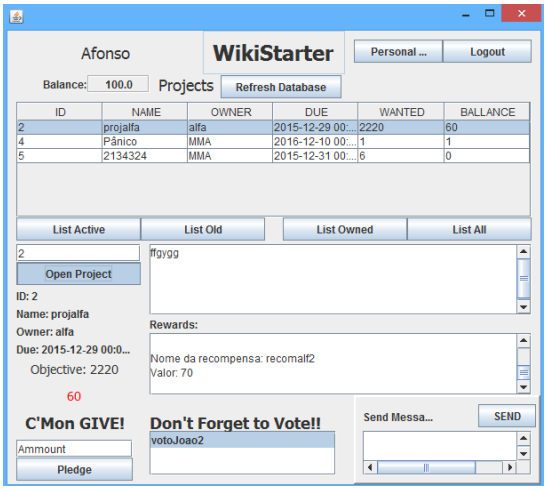
3- Caso o username não esteja na base de dados também é emitida uma mensagem de erro.



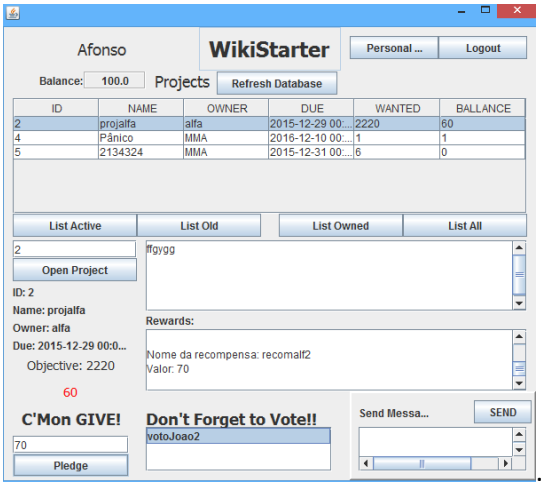
4- Se é aceite o utilizador tem de carregar no botão login para entrar.



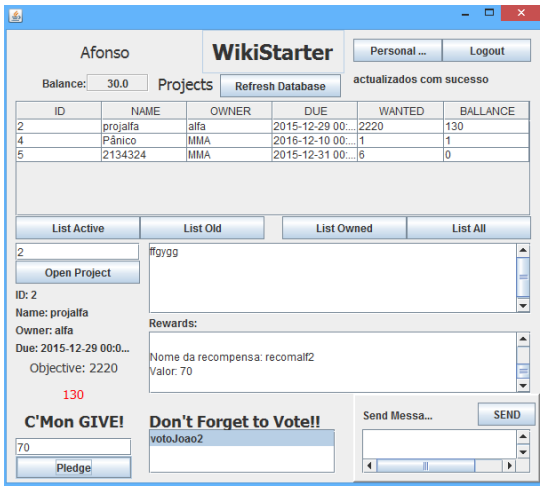
5- É apresentado o seu menu onde estão os projectos com as suas informações, o seu saldo e vários outros campos. Pode listar todos os projectos activos em List Active, os que já passaram a validade em List Old, os projectos dos quais é administrador e podes lista-los todos independentemente da sua categoria. Pode dar Pledge a um projecto, ou seja, doar determinada quantia. Pode ainda votar numa determinada categoria e enviar mensagens.



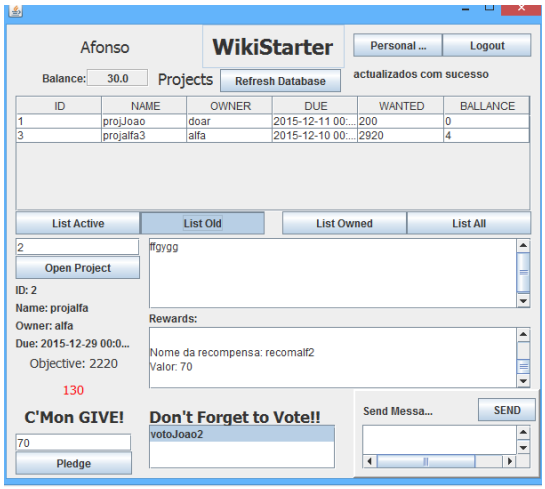
6 – Ao se introduzir o id do projecto, este é aberto e aprecem todas as suas opcções, a sua descrição, pode-se votar no que está disponível, pode-se enviar uma mensagem, etc.



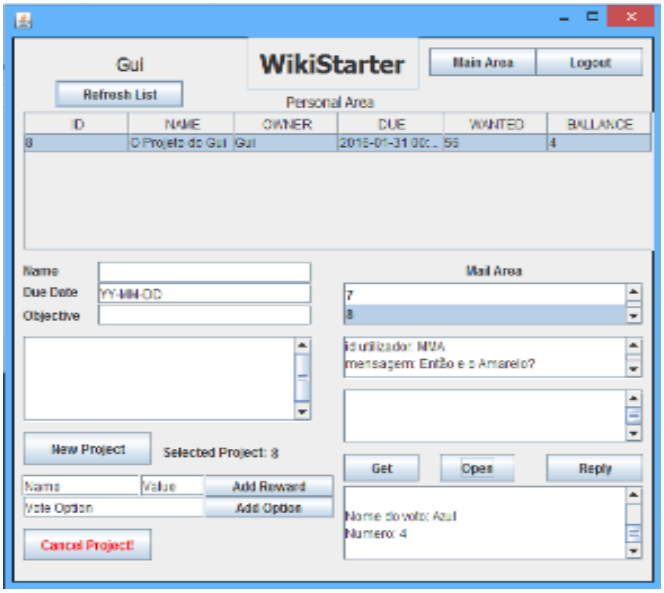
7- O utilizador fez uma doação de 70 ao projecto



8 – O saldo do projecto atualiza, assim como o seu próprio.



9 – Listou os Projectos que já passaram de prazo carregando no “List Old”.



10 – A área pessoa do Utilizador, onde pode criar projectos, ler e responder a mensagens e ver todos os seus projectos com as suas informações. Pode regressar ao menu principal carregando em “Main Area” ou sair fazendo “Logout”.

Manual Programador

- Class sql – Classe onde estabelecemos a ligação com a Oracle, caso não seja possível estabelecer conexão são emitidos os devidos erros.

- setB(String s) e getB(String s) – Funções que nos permitem executar os comandos SQL em Java nas querys.

- void closeDatabase() – Fecha a conexão com a Oracle.

- static synchronized void updateVAL() – Nesta função primeiro começamos por ir atualizar a variável "hoje" em todos os projetos com a data do dia através do SYSDATE. Seguidamente através de código SQL e transações, vamos percorrer os ids todos e comparar a data final do projeto com a data de hoje, se a data final do projeto for inferior ou igual, ou seja já passou de prazo, a validade é passada de 1 para 0. Caso seja detetada alguma exceção durante a transação é feito o rollback que vai desfazer todas as mudanças executadas nessa mesma transação

- static synchronized String verificarValidade() – Primeiramente chama a função anterior para ir fazer update às validade, seguidamente através de SQL vamos verificar os projetos que já passaram de prazo, caso isto aconteça é chamada a função atualizar que vai fazer verificar se os projetos atingiram o saldo pretendido ou não. Se o projeto passou da data com o saldo objetivo então é adicionado na tabela depositos e o EntregueR passa a 1. Se projeto não atingir o saldo objetivo então o EntregueR fica a -1 e não é entregue ao user a recompensa.

- static synchronized void addUser(String uname,String password) – Nesta função pretendemos criar um novo utilizador. Primeiro vamos verificar se o utilizador já existe, caso não exista o user é adicionado à tabela com um saldo no valor de 100.

- static synchronized void login(String uname,String pword) – Nesta função pretendesse apenas realizar um simples login. Caso seja encontrado um utilizador com o nome e a password inserida é retornado que o login foi efetuado com sucesso senão dá uma mensagem de erro;

- synchronized static String depositarSaldo(String user,float valor) – Com esta função pretendemos adicionar uma certa quantia à conta de um user. Para isso, vamos buscar à tabela utilizadores o saldo do utilizador, e adicionamos-lhe o valor do depósito. Seguidamente fazemos update desse depósito numa transação, ou seja, faz update da tabela utilizadores na coluna saldo e volta a ir buscar à tabela o saldo do utilizador e soma o valor do depósito. Se ao fazer select do saldo do utilizador à tabela não ocorrer qualquer tipo de erro, é impressa a mensagem de que foi depositado com sucesso.

- static synchronized void consultarSaldo(String uname) – Esta função apenas vai devolver o saldo de um utilizador caso ele exista através do seu nome à tabela de utilizadores.

- synchronized static String projetosAdmin(String admin) – Primeiramente vai devolver os projetos ativos de um determinado administrador através do seu nome e depois os projetos que já passaram de prazo.

- synchronized static String recompensasUser(String user) – Nesta função através de SQL vamos devolver primeiramente as recompensas de um determinado user, estando as recompensas ainda não validadas pois os projetos ainda estão em decurso. De seguida vamos buscar à tabela as recompensas em curso e as que já foram entregues, caso o utilizador ainda não tenha recebido nenhuma ou não esteja envolvido em nenhuma em curso é devolvida uma mensagem de erro.

- synchronized static String criarProjeto(String nome, String username, String inicio, String fim,float pretendido, float saldo) – Nesta função é pretendido apenas adicionar um novo projeto à tabela projects. Em primeiro lugar vamos verificar quantos projetos é que existem na tabela e adicionamos um, obtendo assim o id do próximo projeto a adicionar. De seguida adicionamos um novo projeto na tabela através de um insert em SQL e atribuímos-lhe o id.

- synchronized static String cancelarProjeto(String uname, int proj_id) – Inicialmente verifica-se que o utilizador é o administrador do projeto, se o projeto não existir é emitida uma mensagem de erro, o mesmo, caso o utilizador não seja o administrador. Caso efetivamente seja, vamos selecionar o utilizador da tabela depósitos através do id do projeto e são

atualizados os saldos dos users (é-lhes devolvido tudo o que doaram) para o projeto ser cancelado. Através de transações, é eliminado o projeto dos depósitos, sendo feito update do estado do projeto, relativamente aos votos, à validade, ao saldo e às recompensas, sendo assim então cancelado o projeto com sucesso.

- synchronized static String atualizar(String uname, int proj_id,int cumprir) – Mais uma vez verifica-se se o user é o administrador do projeto e faz posteriormente update do estado do projeto, relativamente aos votos, validade e recompensas, tendo em conta o facto de se atingiu o não o valor pretendido. Finalmente põe o seu saldo a 0.

- synchronized static String adicionarRecompensas(int idP, String nome, float valor) – Começamos por ir buscar o idP do projeto pretendido ativo e obter o número máximo de ids de recompensas para atribuir o id da próxima a adicionar. Seguidamente verificamos se o nome da recompensa já existe, caso existe é gerada uma mensagem de erro a dizer que esta recompensa já existe, se não existir esta recompensa nova é inserida na tabela e é-lhe atribuído um id igual à soma do id máximo de recompensas mais 1.

- synchronized static String adicionarVotos(int idP, String nome) – Primeiramente vamos verificar se o projeto existe e se se encontra ativo. De seguida, vamos buscar o número máximo de ids dos votos para atribuir o próximo ao voto a adicionar. É feita a verificação de se aquela opção de voto já existe ou não, caso não exista o voto é inserido na tabela de votos com o id do projeto respetivo.

- synchronized static String mostrarRecompensasProj(int id) – Esta função lista todas as recompensas de um projeto que ainda esteja ativo.

- synchronized static String mostrarVotosProj(int id) –

  Lista apenas todos os votos de um projeto com determinado id que ainda esteja ativo. Caso se determine que já passou de prazo devolve uma mensagem de erro.

- synchronized static String listaProjetosActuais() – Lista todos os projetos cuja variável "ativo" está a 1, ou seja, ainda não passou de prazo.

- synchronized String listaProjetosAntigos() – Lista todos os projetos com o "ativo" a 0, ou seja, a sua data final já passou.

- synchronized String listaProjTodos() – Chama apenas as duas funções anteriores.

- synchronized static String doar(String user,int proj,float doacao,String voto) – Começamos por verificar se o projeto a que se pretende doar ainda está ativo e se o user já lhe tinha doado anteriormente. Um user apenas pode doar a um projeto uma vez, caso queira doar outra vez tem de eliminar a doação dos depósitos, sendo o dinheiro que fora doado devolvido ao user e retirado do projeto, voltando assim o user a poder doar. Depois de doar é feito um update ao saldo do utilizador e do projeto, de seguida vamos buscar o id da recompensa e o valor da tabela recompensas do projeto respetivo. Se o valor da doação for superior à recompensa ele passara a ser a pessoa que irá receber essa recompensa. Se o utilizador não tiver um saldo suficiente para fazer a doação é devolvida uma mensagem de erro, se tiver, inserimos o novo depósito na respetiva tabela e fazemos um update aos votos. Ao doar o user escolhe obrigatoriamente uma das opções de escolha (votos) disponíveis no projeto e é atualizado o número de votos dessa escolha tendo em conta a quantia doada (quanto mais, mais o voto vale). No final verificamos se esta doação foi atribuída corretamente.

- static synchronized String mostrarMensagens(int idPro) – Lista todas as mensagens de um projeto. Caso não tenha mensagens é apresentada uma mensagem de erro.

- static synchronized String eliminarDoacao(String user, int idP) – Verificamos se o utilizador doou mesmo ao projeto pretendido e se este ainda está ativo, de seguida fazemos um update do saldo do utilizador que vai voltar a receber a quantia que tinha doado, retira essa quantia do saldo do projeto. De seguida são atualizados os votos e removida a doação da tabela depósitos.

- synchronized static String mandarMsg(String user,int idP,String mensagem) – Começamos por verificar se o projeto prendido e o utilizador existem. De seguida vamos buscar o número máximo de ids de mensagens para atribuir o próximo à mensagem a ser enviada. Adicionando assim a mensagem na tabela com os atributos correspondentes. No final verificamos o número de mensagens e se aumentou. Caso tenha sido enviada é devolvida uma mensagem de sucesso.

- synchronized static String responderMensagens(String user,int idP, int idM, String resposta) – Mais uma vez verificamos se o projeto existe e se o utilizador é o administrador do projeto. Depois fazemos um update às mensagens e adicionamos a resposta com um set.

Diagrama ER Atualizado

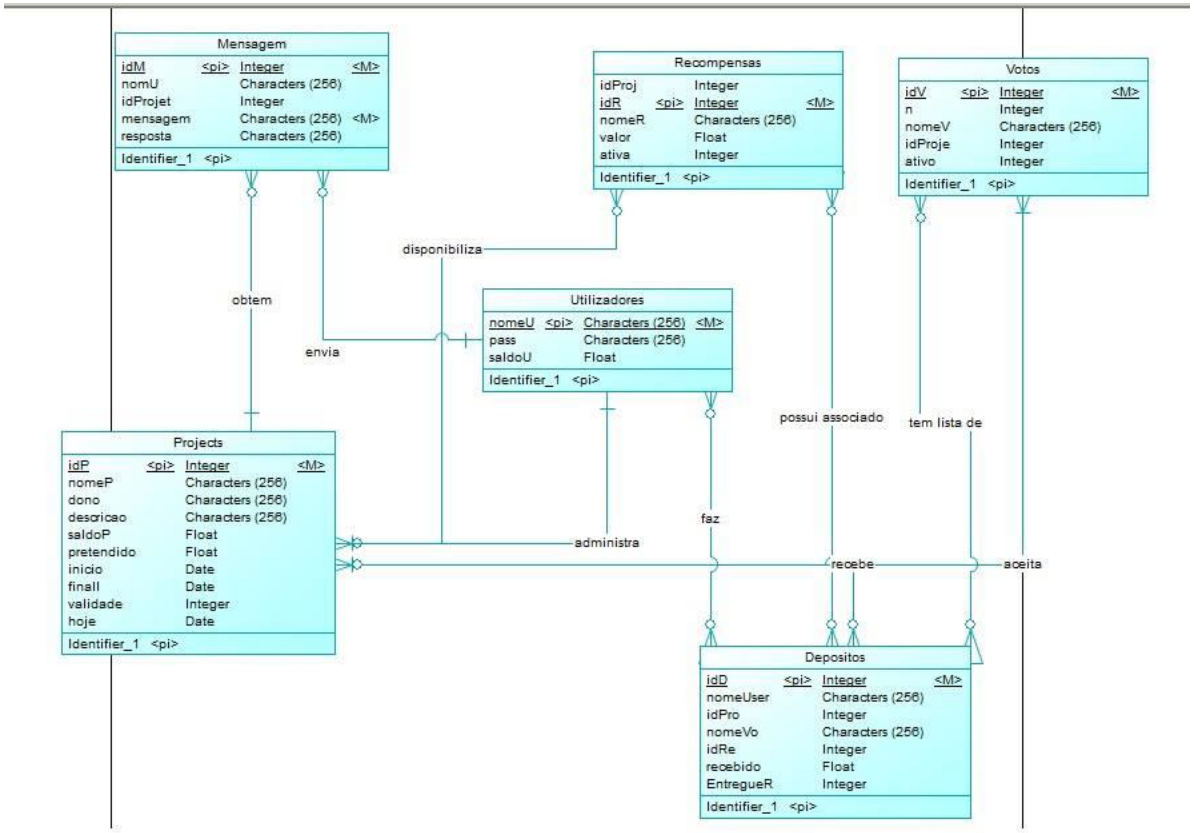
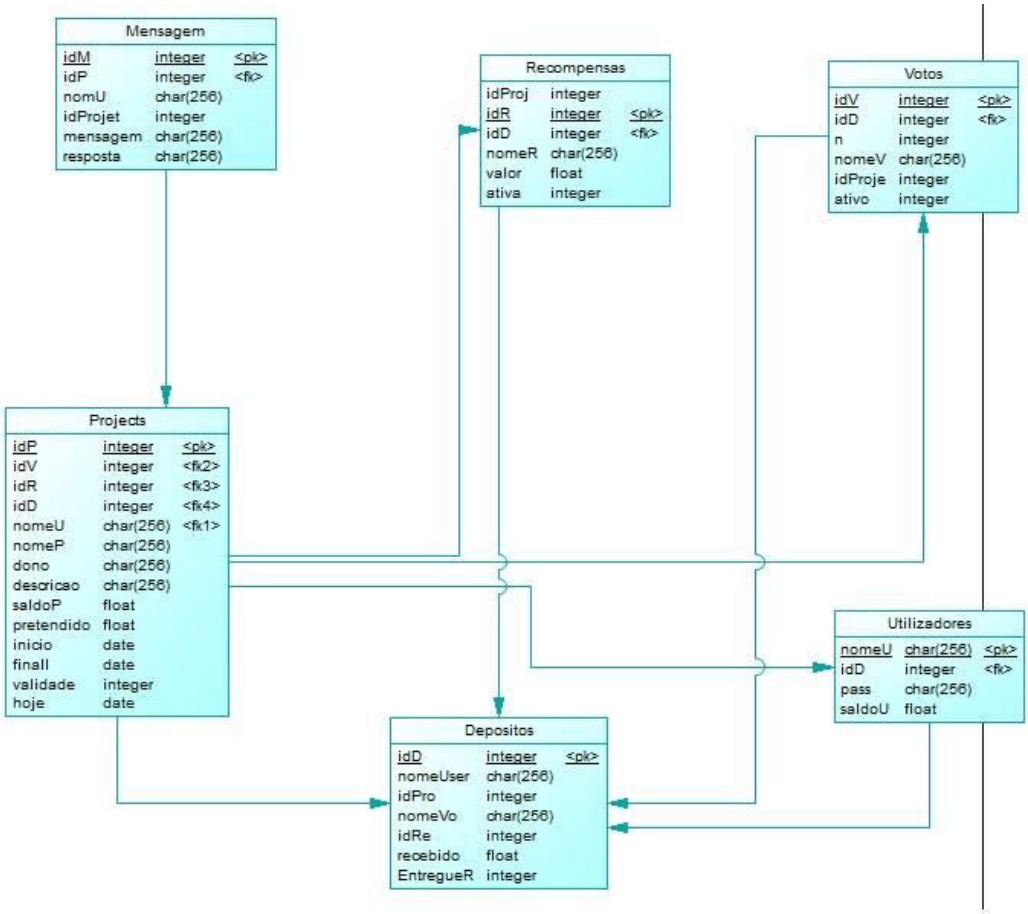


Diagrama Físico Atualizado



Atribuição de Tarefas e Diagrama de Gantt