Universidade Federal de Pernambuco

Centro de Informática

Graduação em Ciência da Computação

Desenvolvimento do sistema web penC

Marina de Meira Lins Haack

Trabalho de Graduação

Recife

Julho de 2016

Universidade Federal de Pernambuco
Centro de Informática

Graduação em Ciência da Computação

Desenvolvimento do sistema web penC

Marina de Meira Lins Haack

Trabalho de graduação apresentado ao Programa de Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação

Orientadora: *Prof. Dr, Patricia Cabral de Azevedo Restelli Tedesco*

Recife
Julho de 2016

Agradecimentos

Primeiramente, gostaria de agradecer ao PET-Informática pela experiência maravilhosa que tive durante estes três anos e seis meses. Graças ao PET, aprendi demais sobre vários aspectos que não vivenciamos na nossa graduação e convivi com pessoas fantásticas que sei que farão a diferença no mundo, do seu próprio jeito. Além disto, fiz amizades e tive experiências que levarei para a vida toda. Em especial Maria Gabriela Cardoso, Leonardo Andrade e Larissa Passos que se tornaram meus melhores amigos e ainda me entrosaram na turma que originalmente não é minha, mas considero como minha turma (Lontra).

Em segundo lugar, gostaria de agradecer à Nathalia Temudo por ter me ensinado e me ajudado tanto no meu primeiro estágio. Ela foi a melhor líder que alguém poderia ter e nunca esquecerei seus ensinamentos.

Gostaria de agradecer também ao Centro de informática como um todo por me proporcionar anos de desafios, muito aprendizado e convívio com professores e pessoas incríveis. À professora Patrícia Tedesco, que me proporcionou a realização deste trabalho de graduação com seu apoio, atenção e conhecimento. E a minha família, por me proporcionar apoio de forma que eu pude me dedicar 100% a minha graduação.

Para as minhas amigas da época do colégio (*Together*) que estiveram comigo por todo esse processo de escolher o curso, passar no vestibular e concluir a graduação bem. Obrigada pelo apoio e por ainda estarmos próximas apesar de tantos anos.

E por último, gostaria de agradecer a mim mesma por ter sido tão forte e por ter me dedicado tanto durante esses anos, mostrando para todos que duvidaram que eu era capaz e sobretudo provando que muitas coisas na vida bastam esforço e dedicação que a pessoa consegue alcançar seus objetivos.

Resumo

Decorrente aos avanços tecnológicos e a ubiquidade da tecnologia na vida das pessoas, a vida humana tem sofrido um impacto significativo no seu dia a dia. Principalmente, procura-se adquirir benefícios desse avanço para facilitar a vivência das pessoas, as quais inclusive hoje possuem a oportunidade de conseguir aprender o que elas desejam sem necessidade de uma escola ou faculdade. A autorregulação tem se mostrado um fator de grande importância na vida destes estudantes, os quais tem estudado cada vez mais fora das salas de aula. Em decorrência dos avanços tecnológicos, surgiram mais oportunidades também para as pessoas aprenderem computação, mesmo não sendo uma tarefa tão fácil. Este trabalho tem como objetivo o desenvolvimento do sistema penC, utilizando como base o modelo apresentado por [SOARES, 2015], respeitando a simplicidade do uso do sistema e de propor melhorias para o mesmo, afim de firmar se o modelo proposto de aprendizado computacional aliado à autorregulação é de fato válido.

Palavras chaves: Autorregulação, tecnologia, penC.

Abstract

Due to technological advances and the ubiquity of technology in people's lives, human life has undergone a significant impact on their daily lives. Mankind is mostly it is looking to acquire benefits of this advance to facilitate the experience of people lives and today, they have the opportunity to get to learn what they want without formal education. Self-regulation has been a very important factor in the lives of these out of classroom students. In consequence of technological advances, there were more opportunities for people to learn computing, even though this is not an easy task. This work aims to develop the penC system using as a base the model presented by [SOARES, 2015], respecting the system's simplicity use and proposing improvements to it, in order to establish if the proposed model of computer learning combined with self-regulation is valid in fact.

Keywords: Self-regulation, technology, penC

Sumário

CAPÍTULO 1

Introdução

Atualmente, é comum pessoas de várias faixas etárias possuírem e utilizarem vários tipos de tecnologias. Tecnologias estas que vão desde celulares, até o uso de uma planilha no trabalho. Entender como funcionam as coisas em termos computacionais pode facilitar o uso das tecnologias e até fornecer os meios de automatizar tarefas manuais exaustivas do dia a dia [MC FARLAND, 2014].

Em atividades relacionadas a programação, as pessoas se envolvem em conceitos da ciência da computação tais como resolução de problemas, lógica de programação, abstração e depuração. Ou seja, estes indivíduos são expostos ao pensamento computacional, que é a utilização destes conceitos da ciência da computação para a resolução de problemas [WING, 2008]. Além do que, o pensamento computacional também está ligado ao desenvolvimento da criatividade e do pensamento crítico, tão importantes em vários campos da ciência [LUDOVICO et al., 2015].

Aprender a programar não é uma tarefa fácil. Programar exige que a pessoa entenda um grande grau de abstração que vai além de dimensões físicas de espaço e tempo [LUDOVICO et al., 2015]. Ela se depara com a escrita formal e não a escrita natural que ela está acostumada, além da linguagem de programação não ser maleável como a escrita natural. Outro problema é que o indivíduo precisa entender o estado em que cada variável está naquele momento, para que o código funcione como desejado [MILLER, 2014]. Todos estes são grandes obstáculos que os estudantes precisam ultrapassar sozinhos para conseguirem aprender a programar.

Hoje, a maioria dos alunos passam mais tempo fora das salas de aulas do que dentro delas e estudos apontam que 70% do ganho dos conhecimentos adquiridos pelos estudantes são feitos fora das salas de aula [TERENZINI et al., 1996]. Pesquisas na área de aprendizagem atestam que os alunos conseguem manejar seu aprendizado e seus resultados [HADWIN, 2011] e os alunos autorregulados se planejam, se auto instruem, se organizam, se auto monitoram e se auto avaliam em vários estágios durante o seu aprendizado [FRANÇA et al, 2015].

Existem várias evidências na literatura sobre os benefícios que a autorregulação traz aos estudantes na hora de estudar computação. Um destes estudos, foi realizado por Bergin el

al [BERGIN et al., 2005] e ele concluiu que estudantes que eram mais autorregulados tiveram melhores desempenhos em relação aos outros. Também existe evidências científicas da utilização da autorregulação com tecnologias como um meio de ajudar os estudantes. Um exemplo disto é o trabalho de Mitrovic, que propôs um ambiente para ajudar a aprender banco de dados com uma estratégia de auto explicação [MITROVIC, 2003].

1.1 Objetivos

O objetivo deste trabalho é desenvolver o sistema penC, baseado no modelo proposto por [SOARES, 2015], respeitando usabilidade e propondo melhorias para o mesmo, afim de firmar se o modelo proposto de aprendizado computacional aliado à autorregulação é de fato válido.

1.2 Estrutura do trabalho

Este trabalho está dividido em 6 capítulos, incluindo este capítulo de introdução. No Capítulo 2, serão abordados com maior profundidade os conceitos aplicados no sistema penC, dando destaque para a autorregulação. Em seguida, o Capítulo 3 discorrerá sobre a análise de competidores realizada para o entendimento do mercado. O Capítulo 4 explicará como o sistema penC foi desenvolvido neste trabalho. Desde a imersão, ideação e prototipação até a explicação do sistema em si. Já o Capítulo 5 apresentará a experimentação realizada afim de comprovar o método e também indicará as melhorias propostas neste trabalho. Por fim, o Capítulo 6 expõe as conclusões obtidas mostrando os resultados da experimentação, implementação realizada, bem como as sugestões de trabalhos futuros.

CAPÍTULO 2

Fundamentação teórica

Neste capítulo serão abordados os conceitos básicos que foram utilizados para a criação da metodologia proposta para o sistema penC.

2.1 Conceitos abordados

A conjunção dos fundamentos da computação com o pensamento crítico define uma metodologia para solucionar problemas nomeada pensamento computacional [WING, 2006]. Esta metodologia explora a visão de que o computador não somente oferece ao indivíduo um novo meio de aprender, como principalmente oferece uma nova forma de aprender a aprender [LU; FLETCHER, 2009]. O pensamento computacional, de forma geral, traz a ideia de utilizar de habilidades comumente utilizadas no desenvolvimento de programas computacionais para resolver problemas nas mais diversas áreas [WING, 2006].

A *Computer Science Teachers Association* (CSTA) em cooperação com a *International Society for Technology in Education* (ISTE) e a *National Science Foundation* (NSF) propôs uma definição operacional do pensamento computacional que expõe quais conhecimentos os alunos devem possuir ao final de sua trajetória na escola. Foi observado pelo estudo realizado por CSTA, ISTE e NSF que o pensamento computacional é um método de resolução de problemas que tendem principalmente a incluir as seguintes características:

- Sistematizar e analisar dados, de modo lógico;
- Escrever problemas de forma que seja possível usar o computador e outras ferramentas para ajudar a resolvê-los;
- Automatizar soluções através do pensamento algorítmico;
- Retratar dados através de abstrações, tais como modelos e simulações;
- Identificar, analisar e implementar as soluções possíveis de forma eficiente e eficaz;
- Escalar o processo de resolução de problemas para uma grande variedade de problemas.

Por conseguinte, estas são a base do pensamento computacional e é recomendado que sejam ensinadas na educação básica de forma progressiva [ISTE; CSTA; NSF, 2011].

A Google, já observando a influência positiva que a metodologia do pensamento computacional traz aos alunos, empenha-se em disseminar esta metodologia nas escolas dos Estados Unidos para alunos do ensino primário e secundário [GOOGLE, 2013]. Para entender melhor como é possível se utilizar do pensamento computacional com problemas do dia a dia, pode-se comparar o método *hashing* da computação com a organização de blocos legos. Explicando mais detalhadamente, o método de *hashing* permite organizar grandes quantidades de dados dividindo em subconjuntos mais gerenciáveis. Este mesmo método pode ser aplicado por crianças para subdividir seus blocos legos por cores, por exemplo, influenciando diretamente o tempo para que a criança encontre uma determinada peça [WING, 2011].

Aprender computação não é uma tarefa fácil e professores tentam facilitar ao máximo a transição dos novos estudantes de tecnologia da informação nas novas disciplinas que eles irão cursar. [BENNEDSEN; CASPERSEN, 2007] afirma que os estudantes possuem dificuldades conceituais em elementos que envolvem o pensamento abstrato e lógico tão presentes na computação. O estudo afirma que tradicionalmente o primeiro ano de cursos de computação sofrem de desistências e notas baixas e os estudantes muitas vezes consideram o curso desinteressante e muito difícil. Neste mesmo estudo é dito que o mau desempenho está diretamente ligado à forma errônea pelo qual os alunos estudam.

2.2 Autorregulação

Existem diversos estudos que apontam que os alunos conseguem ativamente controlar a sua aprendizagem e resultados [HADWIN et al., 2011]. Estudantes ditos auto eficazes e autônomos, ou seja, que planejam, organizam, auto instruem, auto monitoram e auto avaliam em vários estágios durante o próprio estudo são chamados de estudantes autorregulados, nome que vem da aprendizagem autorregulada. Tais estudantes possuem a capacidade de criar ambientes e estruturas que otimizem o seu aprendizado [ZIMMERMAN, 1986].

Em [ZIMMERMAN, 2002], é proposto um modelo de autorregulação cíclico que possui três fases e ajuda os estudantes no seu processo de aprendizagem. A primeira fase envolve o momento do aluno antes de começar a aprender o novo assunto, esta fase é chamada de premeditação e neste instante que o estudante analisa a tarefa e se automotiva. A segunda fase, ou controle volitivo, inicia o processo de executar a tarefa e é nela que os educandos trabalham o autocontrole e a auto-observação. Já a última fase, a autorreflexão, se

dá logo após a tentativa de execução da tarefa. É nela que o aluno se auto avalia comparando resultados anteriores e resultados esperados e analisa sua reação pelo sentimento de autossatisfação em relação ao próprio desempenho.

A base da autorregulação durante o aprendizado, que é a autonomia, competência e eficácia é favorecida através do uso de estratégias de autorregulação. Com isto, o educando consegue obter um ambiente de estudo adequado, definição de metas e planejamento, auto monitoramento, auto avaliação e etc., facilitando assim o aprendizado final do aluno [ZIMMERMAN, 2002]. Assim sendo, é importante formar estudantes que desenvolvam habilidades para gerenciar e regular sua própria aprendizagem.

Na comunidade científica existem pesquisas relacionadas ao uso da autorregulação em disciplinas que envolvem o estudo da computação, como pode ser observado em [BERGIN et al., 2005]. Neste estudo, os resultados alcançados indicam que os estudantes que obtiveram bons resultados utilizaram mais estratégias metacognitivas do que os estudantes com menor desempenho.

CAPÍTULO 3

Ferramentas para o ensino de programação

Neste capítulo será feita uma análise das ferramentas que existem hoje no mercado, comparando-as através da curva de valor.

3.1 Análise das ferramentas

Hoje existem diferentes formas de se aprender a programar que vão desde um professor que ensina em sala de aula até um aluno assistindo a um curso online. Dentre estes métodos, existe o de o aluno utilizar sistemas/plataformas para auxiliar no seu aprendizado assim como o sistema penC. Para entender o que o mercado proporciona hoje, foram analisados dez sistemas/plataformas que ajudam os alunos a aprenderem a programar para comparar com a proposta do sistema penC.

Os critérios escolhidos para a análise dos competidores serão: resolução no próprio sistema, interface amigável e participativa, pré-reflexão, pós-reflexão e avaliação por pares. Estes critérios foram escolhidos pois são partes importantes na metodologia penC desenvolvida em [SOARES, 2015], com exceção de resolução no próprio sistema e interface amigável e participativa que são critérios meramente de usabilidade, todavia são tão importantes quanto os outros.

Estes sistemas foram escolhidos pois são os mais são citados quando se buscam plataformas para auxiliar o aprendizado em programação.

- **Code**

  Esta é uma plataforma que dispõe um auxílio ao aprendizado para diversas faixas etárias, com o mínimo de 4 anos de idade e com resolução no próprio sistema. Eles propõem estratégias de ensino diferentes para cada faixa etária, para que seja possível que os alunos aprendam os conceitos de acordo com seus limites de idade.

  Eles disponibilizam para o estudante a explicação teórica sobre o assunto em questão, com uma interface amigável, além de viabilizar a prática do assunto de modo que o aluno receba *feedback* da plataforma sobre seus erros e acertos da questão. Ele também oferece um pouco de pré-reflexão e pouquíssima pós-reflexão.

- **Scratch**

A proposta desta plataforma é ensinar programação de forma bem simples e diretamente no sistema. O estudante deve construir um jogo simples sobre qualquer tema que o aluno goste. Ele constrói um *script* do jogo utilizando imagens, animações e blocos de programação. Estes blocos seriam comandos em alto nível que devem ser ligados com outros blocos de programação. O estudante constrói o *script* sempre analisando o resultado da sua ação, visualizando o que acontece com o personagem que já existe na tela por *default*.

Além disto, ao finalizar o seu projeto, o aluno pode compartilhar na plataforma o seu jogo e qualquer um pode o jogar ou apenas analisar os *scripts* daquele jogo.

- **Khanacademy**

  Esta plataforma se utiliza da ideia de ensinar aos alunos através de vídeo aulas com uma interface um pouco pesada. O interessado escolhe um tema, cada qual contendo subtemas com várias aulas, em que é ensinado o conteúdo, são dados exemplos de códigos e ao final desses vídeos, o aluno deve fazer desafios, quiz e/ou projetos a respeito deste subtema no próprio sistema. O estudante pode tirar dúvidas na comunidade e pode também solicitar ajuda.

- **The code player**

  A ideia deste site é ensinar os alunos a aprenderem a programar através de vídeos em que uma pessoa ensina um determinado assunto, construindo e explicando o código em questão. Apesar de não possuir uma interface intuitiva ou resolução no próprio sistema como outras plataformas, ele traz uma ideia diferente que é ensinar a partir de exemplos o que agrada uma boa quantidade de estudantes.

- **Code school**

  Apesar deste sistema utilizar da ideia de ensinar programação através de vídeo aulas, ela possui um diferencial que é a *gameficação*. Quando o aluno termina de ver os vídeos, ele precisa resolver desafios corretamente no próprio sistema para ganhar pontos e com estes pontos, o aluno pode assistir a próxima vídeo aula. Estes desafios podem ser perguntas teóricas ou podem ser códigos que o aluno precisa fazer.

- **Code avengers**

  O code avengers propõe que o estudante aprenda através de exercícios no próprio sistema. A plataforma para cada subtema oferece uma série de exercícios, os quais o aluno deve pesquisar como deve fazer e assim tentar resolver. Eles não

disponibilizam nenhum tipo de assunto teórico, ele somente indica exercícios e indica *feedback* sobre as soluções propostas pelo aluno.

- **W3schools**

  Esta plataforma ajuda as pessoas a aprenderem a programar disponibilizando pequenos tutoriais sobre subtemas de um assunto, explicando um pouco da parte teórica e então disponibilizando exemplos de código de modo que o estudante possa alterar e/ou testar este código e analisar o resultado, com uma interface ultrapassada, mas de certa forma fácil de utilizar.

- **Learn how to program**

  A ideia do learn how to program é servir como um guia diário que a pessoa deve seguir para aprender o que deseja. O site, com uma interface pouco amigável, indica uma série de materiais próprios que o aluno deve ler antes mesmo de começar a estudar o que ele quer aprender. Depois disso o site divide o curso por dia e propõe material teórico e prático que o aluno deve fazer naquele dia específico.

- **Codecademy**

  O Codeacademy tem uma proposta de fragmentar um assunto em pequenas partes, disponibilizar uma parte teórica para o estudante ler sobre aquele fragmento e propõe uma parte prática daquele subtema. O sistema, com uma interface fácil de ser utilizada, só deixa a pessoa prosseguir para o próximo passo, quando ela acerta a parte prática do passo atual. Ao final de um conjunto de fragmentos, o aluno é submetido a um teste prático de um grande resumo do que a pessoa aprendeu naqueles fragmentos e quando o estudante o termina, ele ganha uma *badge* por finalizar aquele conjunto.

- **Programae**

  Esta plataforma oferece uma interface amigável e é dividida em duas partes, quero aprender e quero ensinar. Para estas duas opções dadas, o Programae oferece cursos sobre o assunto escolhido e para cada curso, ele oferece para o usuário o curso correspondente de outra plataforma como o code.org e o Codecademy. Ou seja, ele não oferece um conteúdo original para os seus cursos, até o momento.

Para um entendimento melhor da análise de competidores, após analisar estas dez plataformas/sistemas, foi criada uma curva de valor do sistema penC em companhia com os quatro concorrentes que se destacaram na análise como é possível visualizar na figura 1 na próxima seção.

3.2 Curva de valor

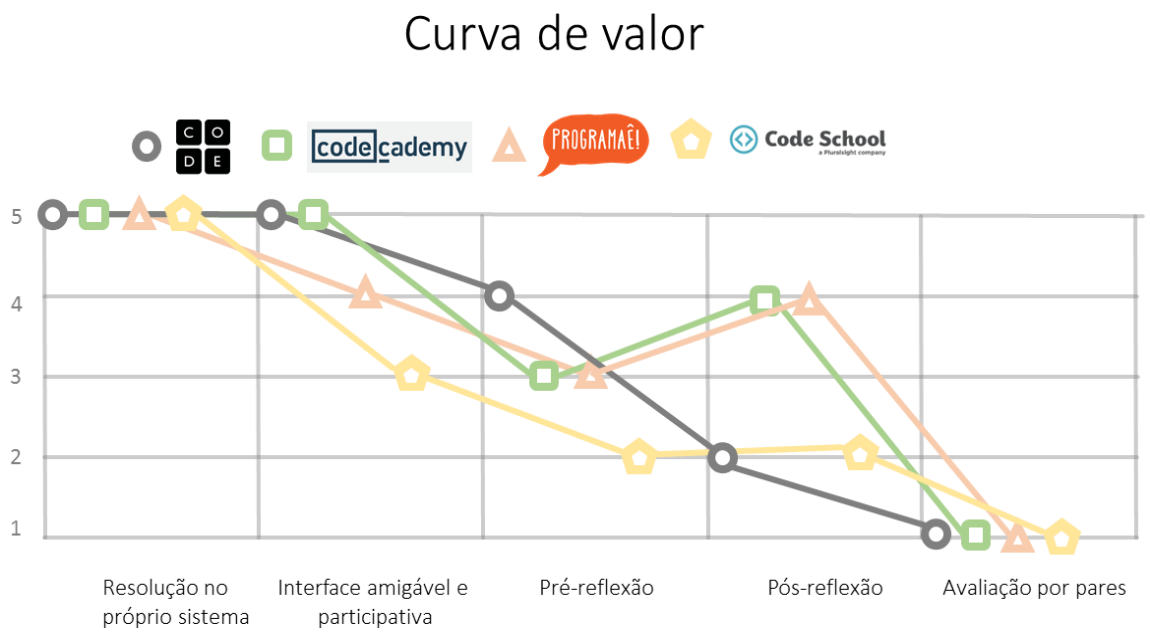


Figura 1: Curva de valor

Uma curva de valor é uma ferramenta comumente utilizada para analisar brechas de mercado para criação de novos produtos pois nela é possível visualizar o diferencial de cada competidor e assim se consegue observar qual o estado atual do mercado [YANG].

Para a construção da curva de valor, foram utilizados os mesmos critérios da análise dos competidores, realizada na seção anterior que foram: resolução no próprio sistema, interface amigável e participativa, pré-reflexão, pós-reflexão e avaliação por pares.

CAPÍTULO 4

Sistema web penC

Neste capítulo será detalhado todo o desenvolvimento do sistema penC, desde o processo de prototipação até a explicação das telas do sistema.

4.1 Prototipação do sistema

Para o desenvolvimento do sistema web penC, foi utilizada a metodologia do *design thinking*, bastante utilizada ao redor do mundo e que vem trazendo resultados bastante significativos, a fim de proporcionar ao usuário final a melhor experiência possível [ADLER et al., 2011]. Essa abordagem é dividida em quatro etapas: i) Imersão, ii) Análise e síntese, iii) Ideação e iv) Prototipação.

Imersão

Esta primeira etapa tem como objetivo o entendimento do contexto do projeto a ser desenvolvido e é dividida em duas partes: Imersão preliminar e imersão em profundidade. A imersão preliminar visa obter um entendimento inicial do escopo e "dos limites" do projeto afim de proporcionar insumos para a imersão em profundidade. A imersão em profundidade possui como objetivo identificar as necessidades dos usuários envolvidos com o projeto e realizar uma investigação do contexto do problema. Esta coleta de dados é realizada através de pesquisa bibliográfica, entrevistas, trabalho de campo, etc [ADLER et al., 2011].

No sistema penC, a fase de imersão preliminar consistiu da leitura de artigos sobre aprendizagem do pensamento computacional, estratégia de autorregulação, necessidades e dificuldades de aprender a programar. Além disso, a dissertação de mestrado da aluna Rozelma Soares de França (autora do modelo do sistema penC) foi lida. Na fase de imersão em profundidade foi realizada uma entrevista com a própria Rozelma Soares, a fim de obter um entendimento melhor do contexto do problema.

Análise e síntese

A segunda etapa visa organizar todas as informações obtidas na etapa anterior com propósito de organizar ideias e apontar padrões que ajudem no entendimento do todo e no reconhecimento das oportunidades e desafios a serem encarados [ADLER et al., 2011].

Em se tratando do desenvolvimento do sistema penC, esta fase correspondeu à análise das informações que existiam acerca da ideia do sistema penC, além do cruzamento de tais informações com as obtidas nos artigos científicos lidos na etapa anterior.

Para concluir esta etapa, foi realizada também uma análise dos competidores a fim de entender o que o mercado já proporciona e como poderiam ser propostas melhorias ao sistema penC com o objetivo de desenvolver um sistema inovador para o mercado atual. Foram analisados doze competidores em relação às etapas da metodologia empregam no penC e em relação à experiência do usuário com a interface do sistema.

Ideação

Esta é a fase na qual os brainstormings[1] acontecem, visando a geração de possíveis soluções para o problema e o perfil do público alvo é definido, ou seja, são definidos os usuários que utilizarão a solução proposta [ADLER et al., 2011].

No contexto do desenvolvimento do sistema penC, esta etapa foi a responsável pelo estudo aprofundado da ideia do penC, afim de entender o público alvo proposto e analisar a ideia proposta em [SOARES, 2015], para então idealizar o sistema.

Prototipação

A última fase é a prototipação, que tem como objetivo transformar as ideias abstratas em um conteúdo formal e material, o protótipo de fidelidade, de modo que seja possível representar e validar a ideia proposta. Existem dois tipos de protótipo, o de baixa fidelidade e o de alta fidelidade. O de baixa fidelidade é um rascunho inicial, com o propósito de realizar validações com o usuário sem um grande gasto de tempo e recursos. Já o de alta fidelidade é o protótipo mais próximo do que vai ser a realidade final do sistema proposto [ADLER et al., 2011]. Geralmente o protótipo de baixa fidelidade é desenvolvido primeiro, para só depois o protótipo de alta fidelidade ser construído.

Para desenvolvimento do sistema penC, já existia um protótipo de baixa fidelidade que foi validado com alguns usuários em [SOARES, 2015], e que foi utilizado como base para os outros protótipos. Com este protótipo de baixa fidelidade pronto, foram desenvolvidos protótipos de alta fidelidade de forma bem sistemática e separada entre as interfaces na visão do professor e na visão do aluno. O passo a passo realizado foi:

1) Desenvolvimento do protótipo de alta fidelidade

[1] Brainstormings: Técnica de criatividade em grupo para encontrar soluções para um problema.

2) Validação do protótipo de alta fidelidade
3) Repetição do ciclo anterior

Para o desenvolvimento da primeira versão deste protótipo, já foram acrescentadas melhorias de usabilidade e design em relação ao protótipo de baixa fidelidade proposto em [SOARES, 2015]. O protótipo proposto possuía um *design* fora dos padrões propostos pelo *material design lite (MDL)* do Google, enquanto que o protótipo de alta fidelidade utiliza este padrão.

O MDL é um guia desenvolvido pelo Google afim de criar um padrão para as interfaces gráficas da Web e permitir uma experiência do usuário unificada entre diferentes dispositivos. Para que a criação deste guia fosse possível, houve um grande estudo de combinação das cores, simetria, posicionamento dos objetos, entre outros aspectos, para proporcionar ao usuário a melhor experiência possível [MATERIAL DESIGN, 2015].

Para obter uma experiência melhor para os usuários que iriam validar o sistema, o Power Point foi utilizado como ferramenta para a criação dos protótipos de alta fidelidade. Com ele é possível conectar as telas e criar ações de forma a torna-las próximas da realidade do sistema. Após finalização da primeira versão do protótipo de alta fidelidade, foi realizada uma validação com possíveis usuários do sistema. Para padronizar a validação e tornar possível a extração do máximo de informações possíveis, foi utilizado um roteiro para a validação utilizando a técnica de *mockups* [ADLER et al., 2011]. Essa técnica tem como objetivo simular a utilização do sistema por parte do usuário de forma não só a testar a posição dos recursos na tela, mas também o fluxo entre as telas.

Como explicado anteriormente, a validação foi dividida em duas partes: Validação das telas na visão do aluno e validação das telas na visão do professor. Para a visão do aluno, foram realizadas duas iterações de validações, gerando três versões do sistema. Já para a visão do professor, foi realizada uma iteração de validações, gerando duas versões do sistema.

Os roteiros utilizados para as validações com os usuários:

- **Primeira versão do sistema visão do aluno:**
    1. Clique em qualquer opção do menu
    2. Veja suas *badges*
    3. Tente resolver qualquer problema
    4. O que entendeu da tela após clicar em resolver (problema)?
    5. O que entendeu da tela após clicar em tentar resolver problema?
    6. Avalie uma solução de problemas

7. O que entendeu da tela após clicar em avaliar solução?
8. O que entendeu da tela após clicar em tentar avaliar solução?
9. Veja seu resultado
10. O que entendeu da tela após clicar em ver resultado e refletir?
11. Veja as discussões
12. O que entendeu da tela após clicar em discussões?
13. Veja uma solução

- **Segunda versão do sistema visão do aluno:**
    1. Veja suas *badges*
    1. Tente resolver qualquer problema
    2. O que entendeu da tela após clicar em resolver (problema)?
    3. O que entendeu da tela após clicar em tentar resolver problema?
    4. Avalie uma solução de problemas
    5. O que entendeu da tela após clicar em avaliar solução?
    6. O que entendeu da tela após clicar em tentar avaliar solução?
    7. Veja seu resultado
    8. O que entendeu da tela após clicar em ver resultado e refletir?
    9. Veja as discussões
    10. O que entendeu da tela após clicar em discussões?
- **Primeira versão do sistema visão do professor:**
    1. Clique em qualquer opção do menu
    2. Clique em gerenciar *badges*
    3. Crie um problema
    4. O que entendeu de cada aba de criar problema?
    5. Clique em acompanhar aprendizagem
    6. O que entendeu da tela de acompanhar aprendizagem?
    7. Clique em medir discussão
    8. O que entendeu da tela de discussão?

Graças a essas validações com os usuários, foram obtidas várias informações que foram utilizadas para melhorar o sistema. Estas melhorias incluem, embora não estejam limitadas a: fluxo das telas, posicionamento de componentes e mudança de componentes. As figuras 2 a 6 correspondem a evolução de algumas telas do protótipo de alta fidelidade na visão do aluno.

- **Visão do aluno:**

Figura 2: Protótipo versão 1

Figura 3: Protótipo versão 2

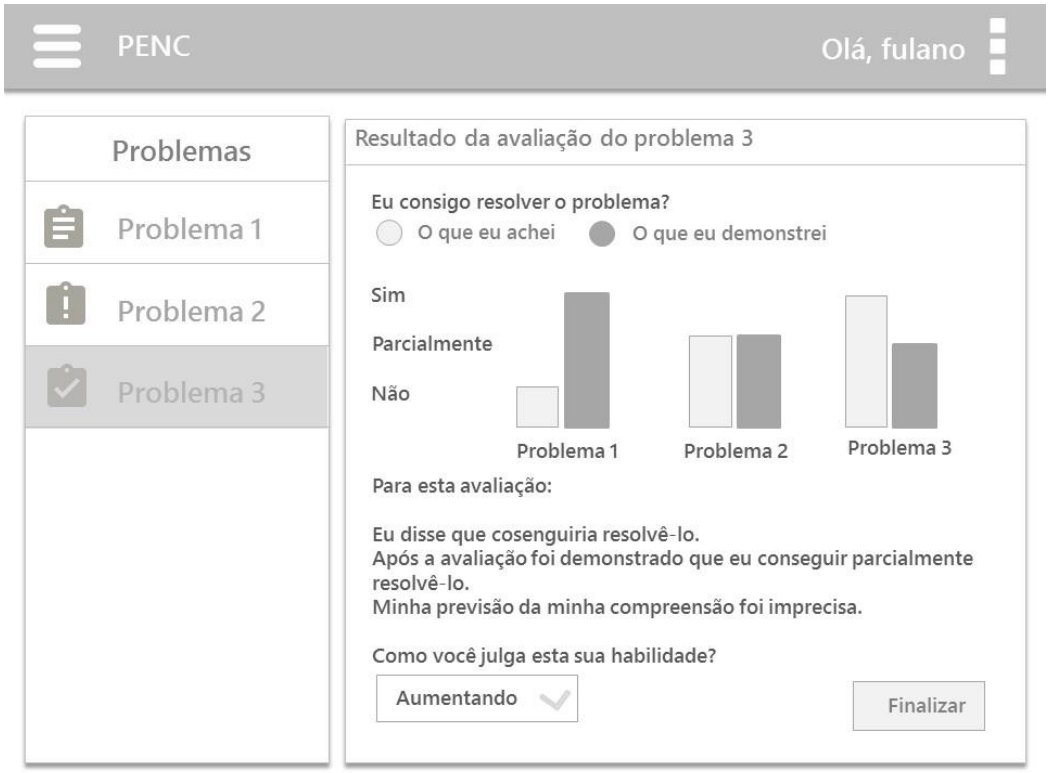


Figura 4: Protótipo versão 2

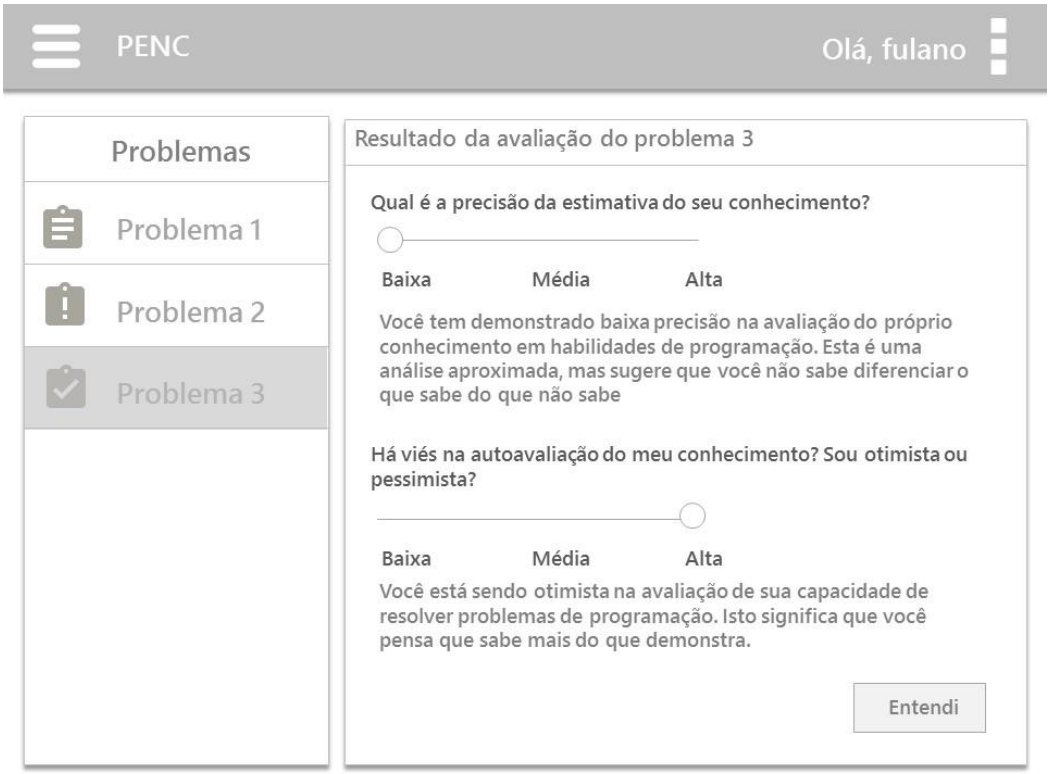


Figura 5: Protótipo versão 2

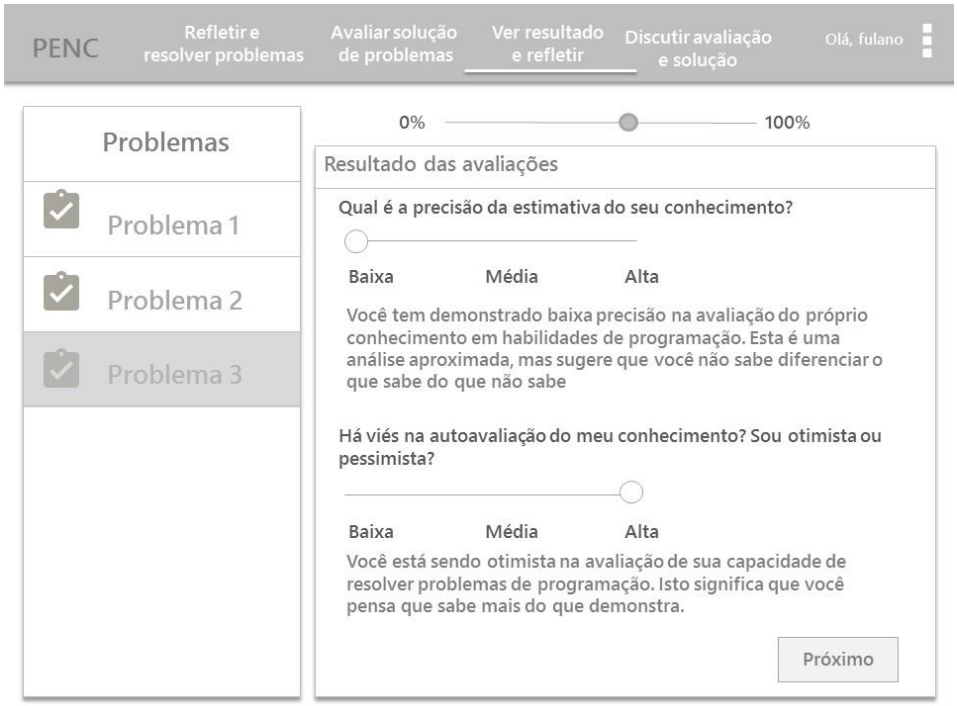


Figura 6: Protótipo versão 3

O primeiro aspecto da evolução dos protótipos é o menu. Na figura 2, o menu só aparecia quando clicado no ícone ao lado e a partir da figura 3 ele sempre é mostrado. Muitos usuários reclamaram pois não sabiam em que parte do sistema estavam já que o menu sempre estava escondido.

Um outro aspecto importante foi identificado a tela de ver resultado e refletir, pois, na primeira versão (figuras 4 e 5) era muito confuso para o usuário a reflexão, já que ele não tinha como identificar em qual estágio estava. Na nova versão (figura 6) pode-se notar que fica claro para o usuário em que estágio da reflexão ele está.

- **Visão do professor:**

O mesmo problema do menu foi observado também na visão do professor, e foi corrigido na versão seguinte do protótipo. A seguir serão exibidas as figuras 7 e 8, que mostram um pouco da evolução destes protótipos.

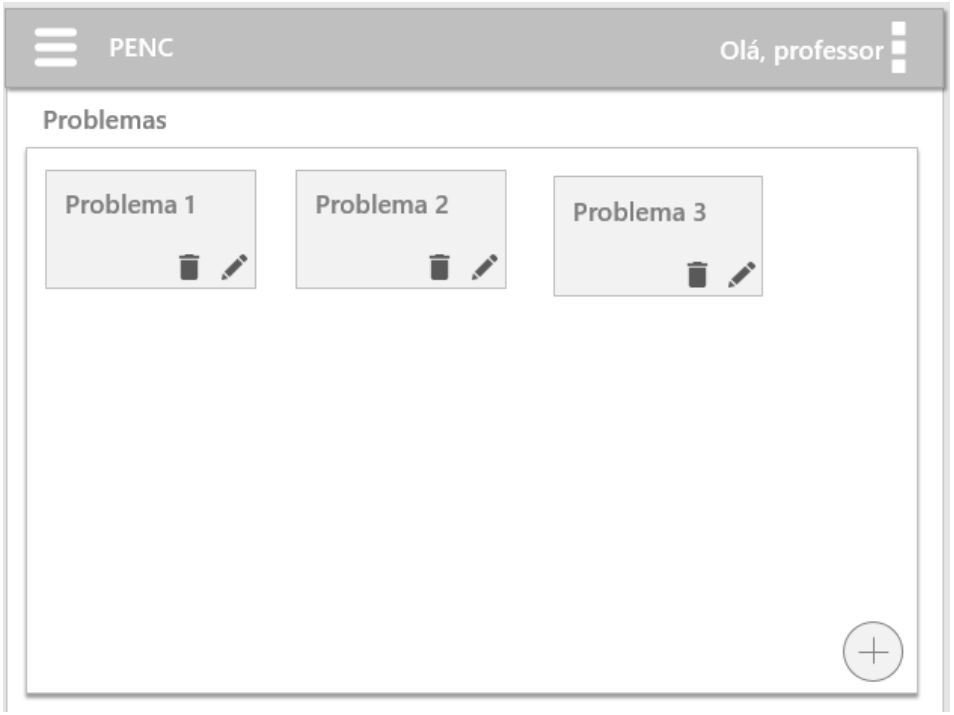


Figura 7: Protótipo versão 1

Figura 8: Protótipo versão 2

Um problema observado na figura 7 é que o professor não sabia instantaneamente à qual turma aquele problema correspondia. Para tal, o mesmo tinha que abrir o problema para saber se era o que ele queria alterar ou deletar. Então, para facilitar, os problemas foram divididos em classes como é observado na figura 8.

4.2 Visão geral

Após a realização das validações, o desenvolvimento do sistema web foi iniciado utilizando as seguintes tecnologias:

- C# como a linguagem principal do sistema;
- HTML, CSS e jQuery para o desenvolvimento das telas;
- SQL Server para o banco de dados;
- Entity framework para a integração do banco com as telas.

Na figura 9 é definida a visão geral da arquitetura do sistema.

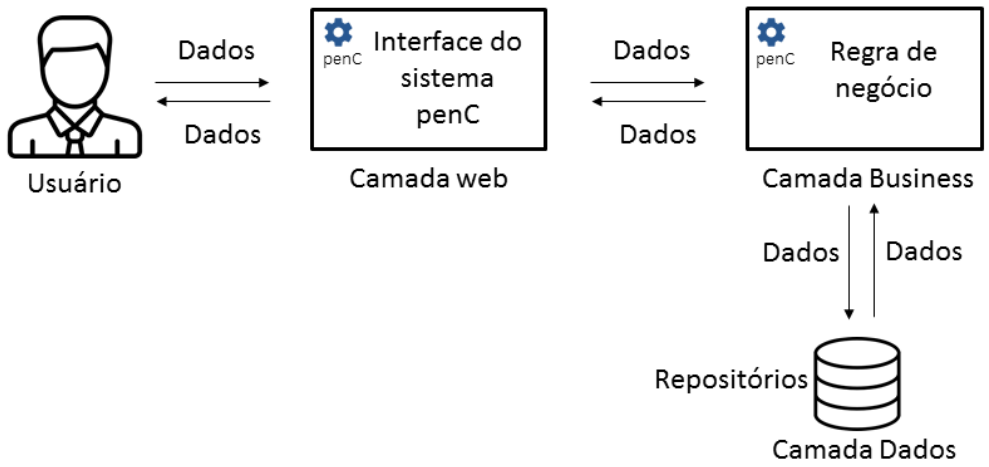


Figura 9: Visão geral da arquitetura do sistema

Como é possível notar pela figura 9, o projeto foi dividido em três camadas:

1. **A camada *Web***

Esta é a camada responsável por toda comunicação do usuário com o sistema. É nesta camada que existe o controle dos dados que serão enviados pelo usuário para o sistema e vice-versa, ou seja, é nela que todas as telas do sistema ficam. Para esta camada foi utilizado o conceito de MVC (*Model-view-controller*) como padrão de arquitetura de software. Portanto, a camada web foi dividida em model (responsável por ser o esqueleto da

informação a ser colocada na tela), *view* (responsável pelos códigos HTML, ou seja, todo o design da tela) e *controller* (responsável por tratar os dados que serão enviados para a tela e os dados que serão recebidos da tela). Nesta camada também estão presentes os arquivos javascript e CSS. Ela se comunica com a camada *business*.

2. **A camada *Business***

Esta camada é a responsável pelas regras de negócio do sistema, ou seja, é nela que são realizados tratamentos mais complexos dos dados para serem salvos no banco de dados ou para serem enviados para a camada *Web*. Por exemplo, quando um professor finaliza um problema, é esta camada que fica responsável por criar os dados relacionados à solução, à revisão e à reflexão para cada aluno da turma especificada no problema. Esta camada se comunica com a camada *web* e a camada *data*.

3. **A camada *Data***

Responsável por representar os repositórios do sistema, é a camada que se conecta com o banco de dados, então ela é encarregada de salvar e obter os dados no mesmo. Estes dados são obtidos através de instruções em alto nível que o *Entity Framework* transforma em instruções SQL puras. Em seguida, os dados são transformados em objetos representados pelos *models* que são utilizados pelas *views* como explicado anteriormente. Esta camada se comunica com a camada *business*.

Database

Este é um projeto existente na solução responsável por todo o gerenciamento do banco de dados em si. Então nele existe uma representação das tabelas e scripts de pré e pós criação do banco de dados. Ele também dá suporte ao incremento ou criação do banco de dados automático, facilitando o desenvolvimento.

4.3 Apresentação do sistema

O sistema web penC é dividido em duas grandes partes: a visão do aluno e a visão do professor. Portanto, para um melhor entendimento, o sistema será apresentado em duas partes.

Visão do professor no sistema:

O sistema na visão do professor possui, a grosso modo, as funcionalidades necessárias para o professor criar problemas e acompanhar o aprendizado dos alunos, já que a ideia

proposta é que o aluno deve sempre estar refletindo sozinho, trabalhando a autorregulação. O professor possui as seguintes funções no sistema:

- Gerenciar turmas

Nesta tela o professor pode criar, editar e deletar quantas turmas ele quiser. Desta forma ele não precisa se limitar a uma única turma no sistema.

- Problematizar

Nesta tela o professor pode criar vários problemas, de forma que enquanto ele não finalizar o problema, não é definido com qual turma o problema está relacionado. Além disto, somente neste estado (de não finalizado) o professor pode editar ou deletar o problema. Uma última funcionalidade desta tela é a de compartilhar a questão com outros professores que estão cadastrados no sistema. Esta funcionalidade surgiu com a ideia de trazer para o sistema o uso do recurso educacional aberto (REA). O REA tem como objetivo englobar o conceito de objeto de aprendizagem, ressaltando o conceito de abertura além de envolver o ato de direito de uso, reuso, revisão, *remix*, redistribuição e adaptação [AMIEL, 2011]. O funcionamento do compartilhamento de problemas será melhor detalhado na funcionalidade compartilhamento de problemas. Na figura 10 é possível ter uma visão geral da tela de problematização.

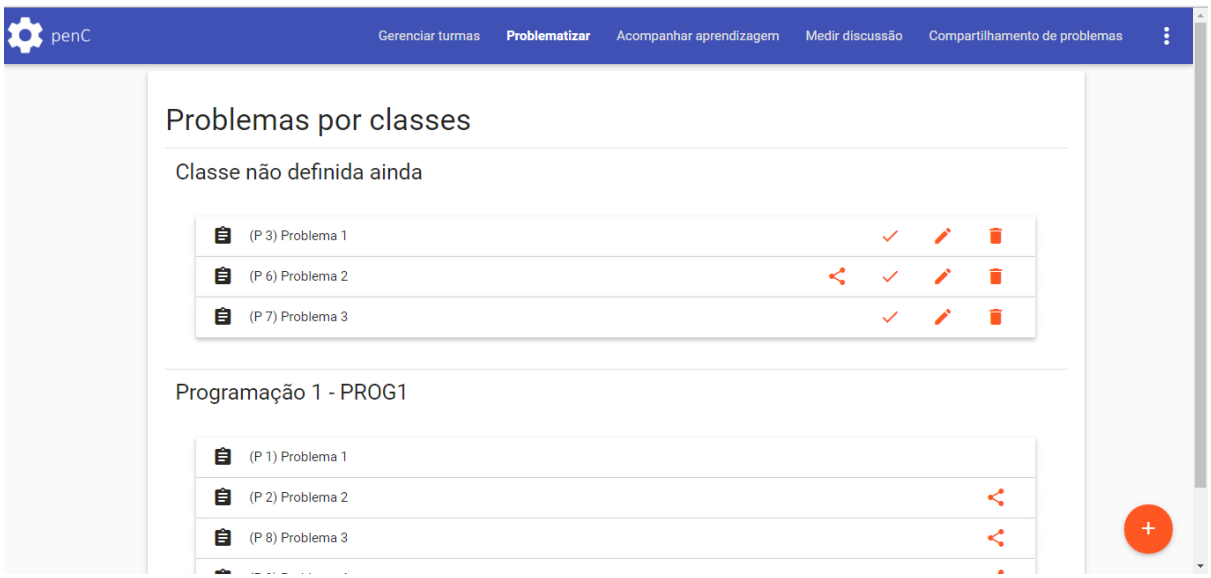


Figura 10: Tela de problematizar

- Acompanhar aprendizagem

Esta funcionalidade proporciona ao professor um meio de acompanhar o aprendizado dos alunos durante todo o curso. É possível acompanhar a solução e reflexão do

aluno para cada problema, além da revisão feita pelo revisor da questão. Também é mostrado um comparativo em forma de gráfico da nota que o aluno acha que vai tirar na questão, do quanto ele realmente tirou, da média em relação as outras questões desenvolvidas pelo aluno e a média em relação a turma. A figura 11 mostra a visão geral da tela de acompanhar aprendizagem.

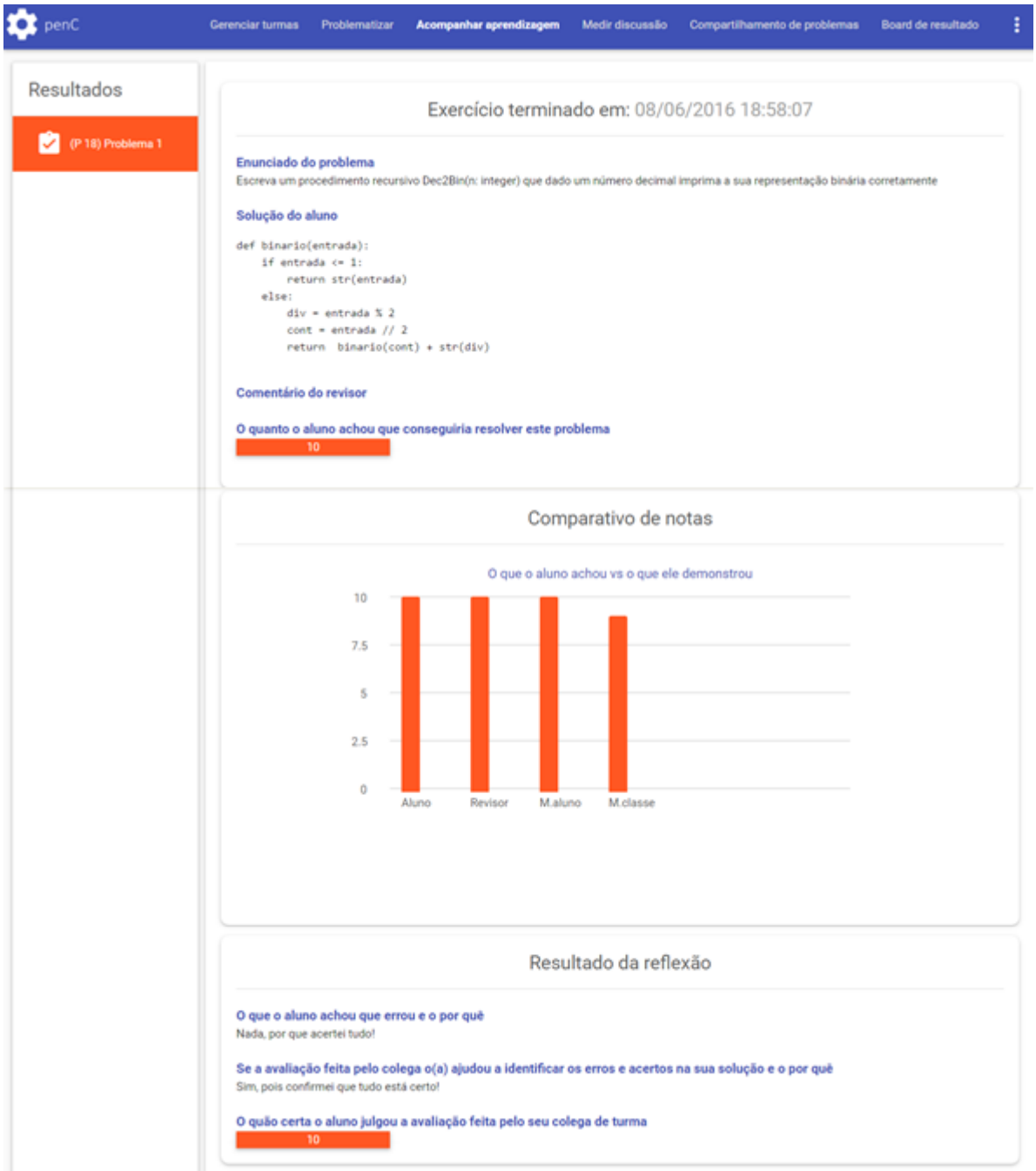


Figura 11: Tela de acompanhar aprendizagem

- Mediar discussão

Nesta tela, alunos e professores podem discutir as questões compartilhadas pelos alunos, proporcionando um momento para debater em grupo sobre possíveis soluções, impactos de custo computacional, etc. Na figura 12, é possível ver a tela de mediar discussão.

Figura 12: Tela de medir discussão

- Compartilhamento de problemas

É uma funcionalidade que permite aos professores compartilharem suas questões com outros professores, trazendo a ideia de REA como comentado anteriormente. É possível que os professores baixem essas questões para o seu repositório de problemas, além de permitir que eles editem o quanto quiserem o problema baixado. Na figura 13, é possível ver a tela de compartilhamento de problemas.

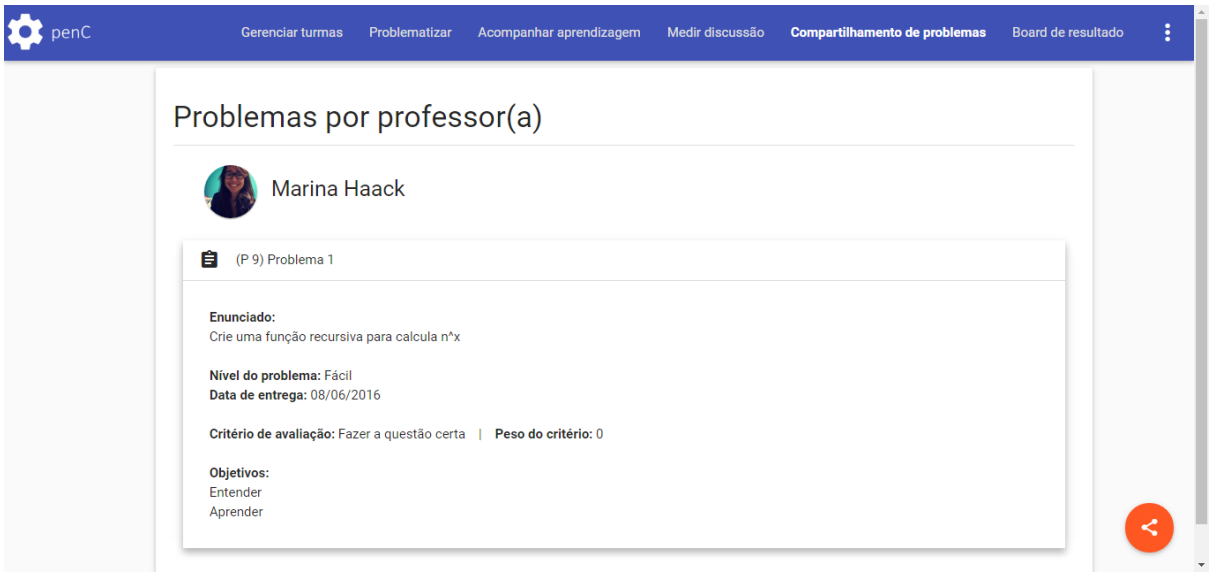


Figura 13: Tela de compartilhamento de problemas

- *Board* de resultado

Por se tratar de um sistema *gameficado*, em que os alunos ganham pontos por conquista de *badges*, resoluções de problemas, revisões feitas, problemas refletidos e objetivos acertados, é necessário expor aos alunos e professores pode retirar o rendimento dos alunos. Na figura 12, é possível ver a tela da *board* de resultado.

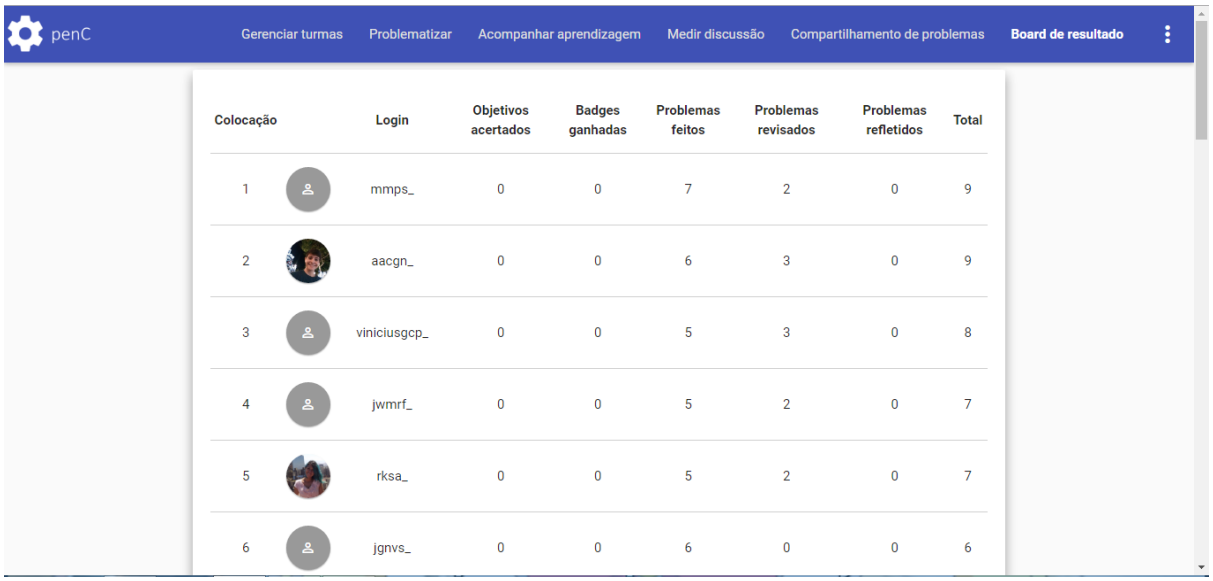


Figura 14: Tela de board de resultado

- Suas *badges*

Como comentado anteriormente, o penC é um sistema *gameficado*. Para proporcionar ao professor formas configuráveis de distribuir pontos a seus estudantes, foi desenvolvida esta tela de *badges*. Nela, o professor cria *badges* que os alunos possam ganhar e ele escolhe os requisitos para tal. Os requisitos são: o quanto o aluno precisa ganhar na revisão, até quantos dias antes do prazo final ele precisa terminar a questão, em quanto tempo o aluno precisa terminar depois que o professor compartilhou a questão e se o aluno precisa ter sido o primeiro a finalizar o problema. Na figura 15, é possível ver a tela de *badges*.

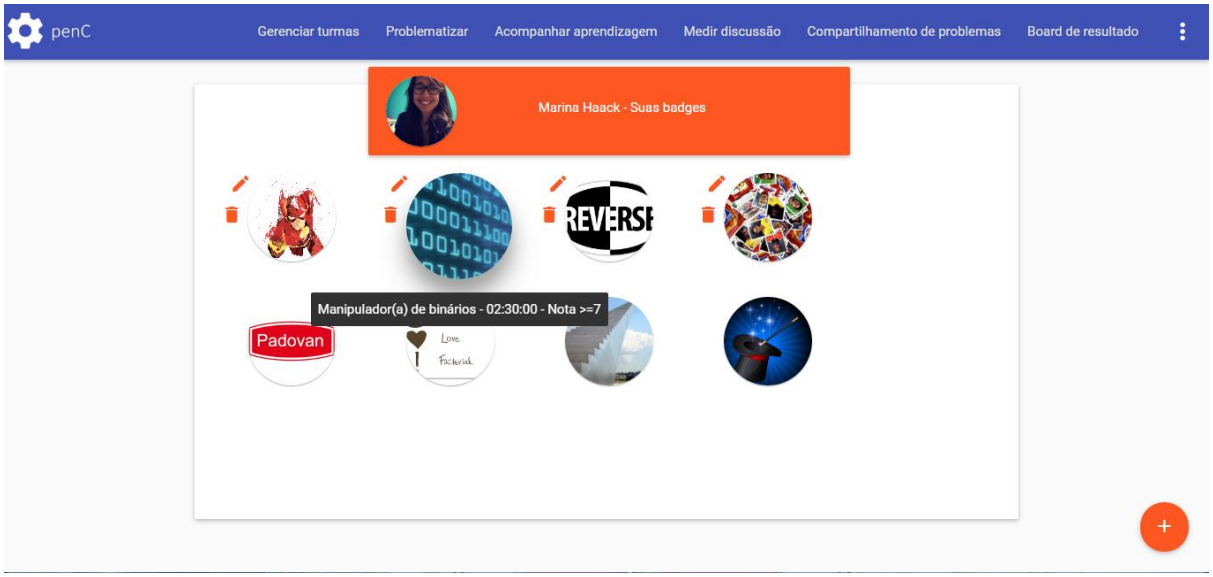


Figura 15: Tela de suas badges

Visão do aluno no sistema:

O sistema na visão do aluno proporciona o estudante as funcionalidades, a grosso modo, de resolver, revisar, refletir e discutir problemas. Detalhadamente o aluno possui as seguintes funções no sistema:

- Refletir e resolver problemas

Esta é a parte do sistema que o aluno tem que refletir e resolver os problemas propostos pelo professor. Para cada problema, o aluno A vai gerar uma solução que será revisada pelo aluno B e em seguida, o aluno A irá refletir acerca da sua própria solução e a revisão que o aluno B fez. O aluno é notificado quando um novo problema pode ser resolvido, quando ele já terminou e quando ele está atrasado para responder o problema. Cada questão tem um prazo associado e o aluno pode ver que *badge* (e seu respectivo requisito) ele pode ganhar por resolver e refletir esta questão.

Antes de o aluno começar a responder uma questão, ele irá indicar o quão difícil ele acha que vai ser resolver esta questão e o quanto ele acha que vai conseguir fazer desta questão. Quando ele terminar de refletir a cerca deste problema, o aluno está apto a responder à questão. Na figura 16, é possível ver a tela de refletir e resolver problemas.

Figura 16: Tela de refletir e resolver problemas

- Avaliar solução de problemas

Esta é a tela na qual o aluno irá revisar as soluções de outros alunos. Antes de o indivíduo analisar a resposta do colega de turma, ele informa o quanto ele entendeu dos critérios de avaliação que o professor indicou para aquele problema. Depois de responder acerca dos critérios de avaliação, o estudante analisa a resposta do colega e informa que nota ele daria para a solução e também informa os erros encontrados na solução, se presentes. Na figura 17, é possível ver a tela de avaliar solução de problemas.

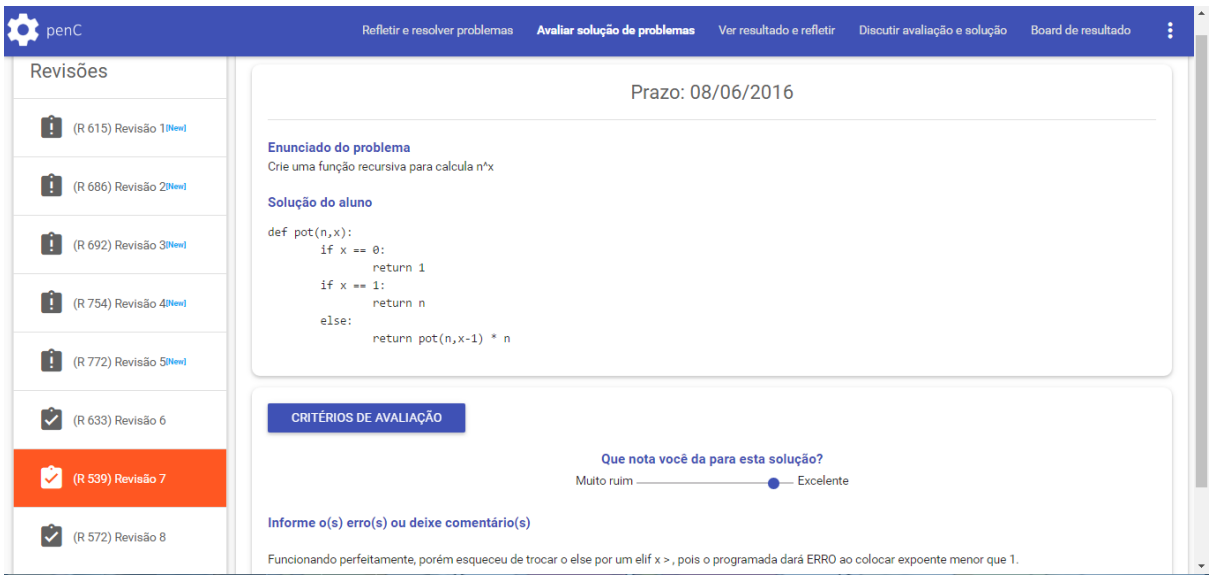


Figura 17: Tela de avaliar solução e problemas

- Ver resultado e refletir

Este é o momento no qual o aluno reflete acerca de sua solução, juntamente com a revisão feita por outro aluno. Neste processo ele informa o que ele entendeu estar errado na solução dele, se cabível, informa se a avaliação feita pelo colega o ajudou a identificar os erros e acertos na sua solução e o porquê e também informa o quão certa ele julgou a avaliação feita pelo seu colega de turma. Após finalizar esta parte, o aluno tem um gráfico informando o quanto ele achou que conseguiria fazer da questão e o quanto o colega achou que ele merecia pela solução dele. Na figura 18, é possível ver a tela de ver resultado e refletir.

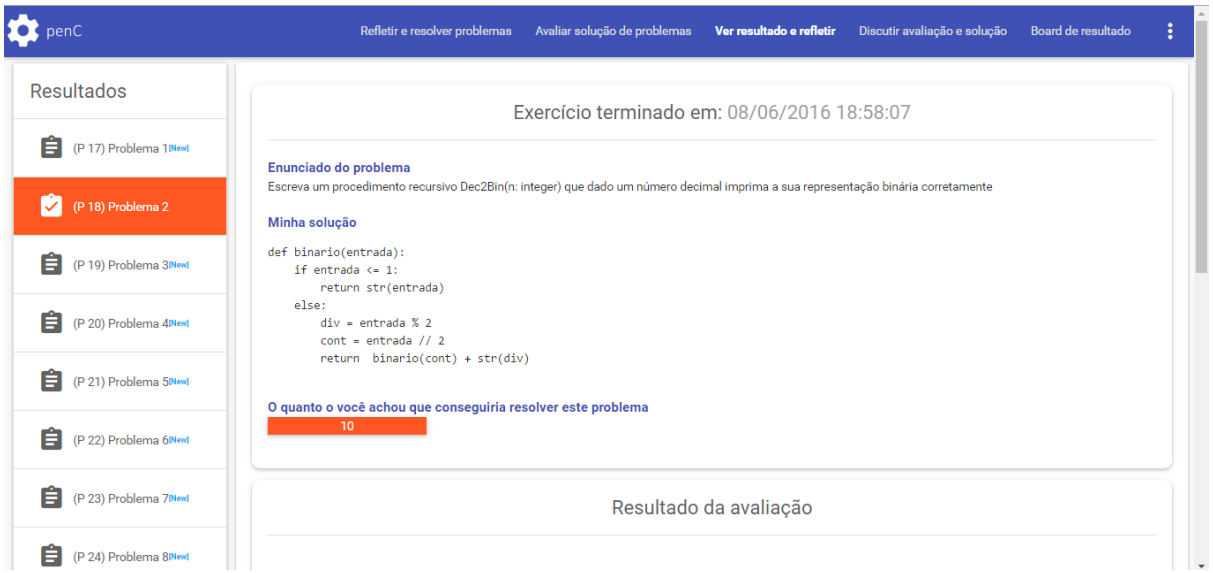


Figura 18: Tela de ver resultado e refletir

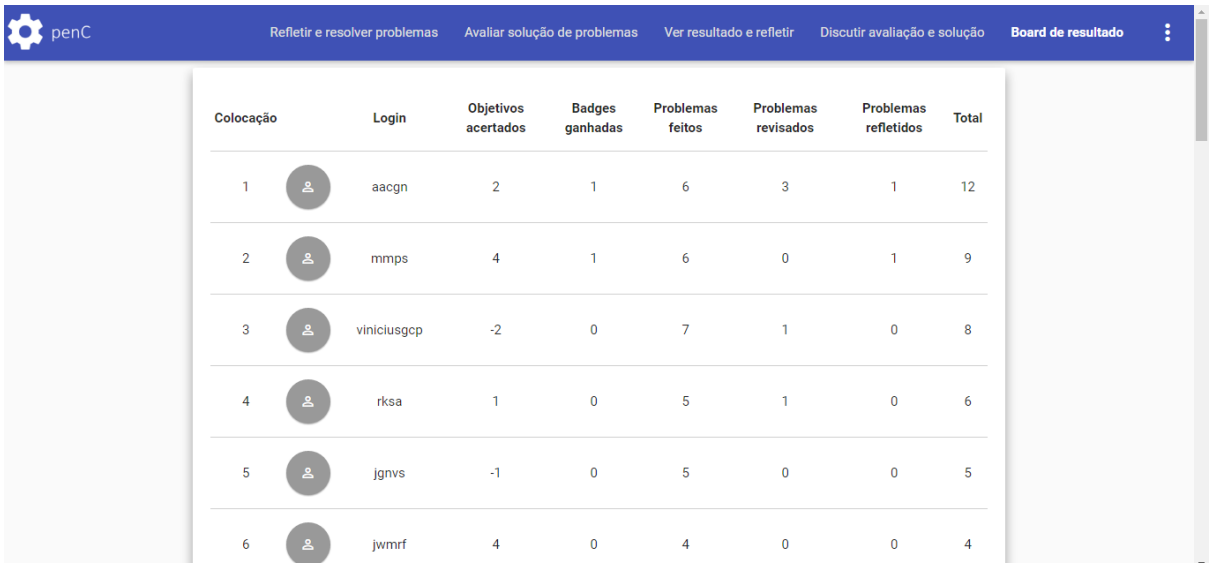
- Discutir avaliação e solução

Este é um requisito compartilhado com o professor no qual os alunos podem discutir acerca de uma solução, como comentado anteriormente na visão do professor do sistema. Na figura 19, é possível ver a tela de discutir avaliação e solução.

Figura 19: Tela de discutir avaliação e solução

- *Board* de resultado

Esta também é um requisito compartilhado com o professor, no qual o aluno pode acompanhar a sua pontuação e a dos seus colegas de turma, como comentado anteriormente na visão do professor do sistema. Na figura 20, é possível ver a tela de *board* de resultado.



Colocação	Login	Objetivos acertados	Badges ganhadas	Problemas feitos	Problemas revisados	Problemas refletidos	Total
1	aacgn	2	1	6	3	1	12
2	mmps	4	1	6	0	1	9
3	viniciusgcp	-2	0	7	1	0	8
4	rksa	1	0	5	1	0	6
5	jgnvs	-1	0	5	0	0	5
6	jwmrf	4	0	4	0	0	4

Figura 20: Tela de board de resultado

- Minhas *badges*

Aqui o aluno pode checar as *badges* que ele possui. Ao passar o mouse por cima da *badge*, o educando consegue visualizar os requisitos que ele alcançou para conseguir aquela *badge*. Na figura 21, é possível ver a tela de *badges*.

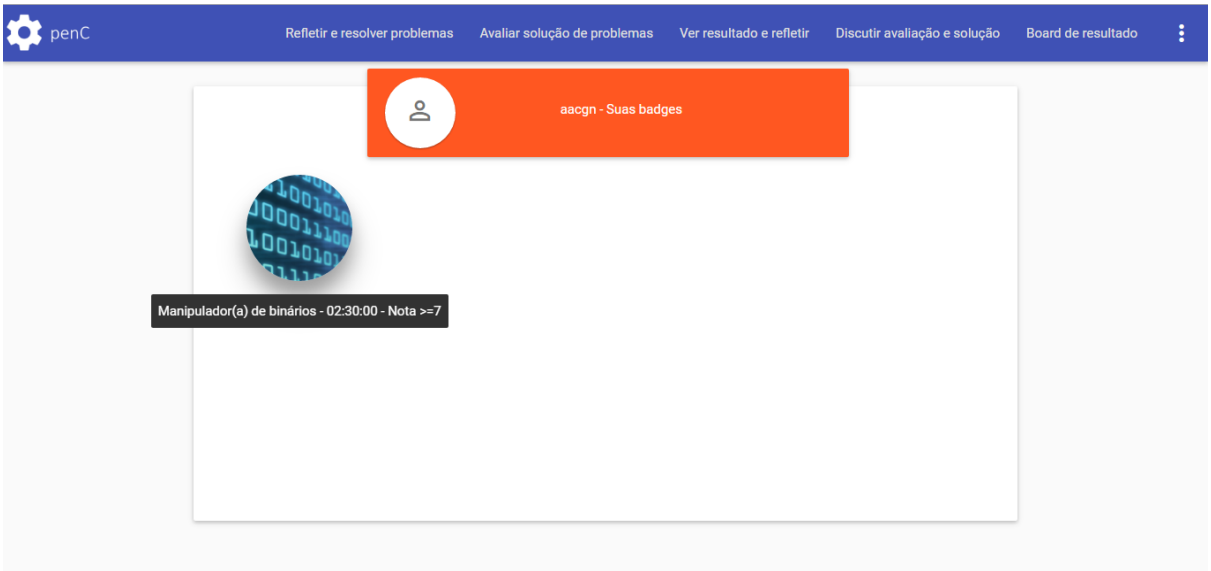


Figura 21: Tela de suas badges

Uma outra função que não é possível ser observada nas telas é a de enviar e-mail. Quando um professor compartilha com os alunos uma nova questão, o sistema envia automaticamente um e-mail para todos os alunos daquela turma específica que um novo problema foi criado e está disponível para ser resolvido. Na figura 22, é possível ver o e-mail enviado pelo próprio sistema para indicar que foi compartilhado um novo problema com o aluno.

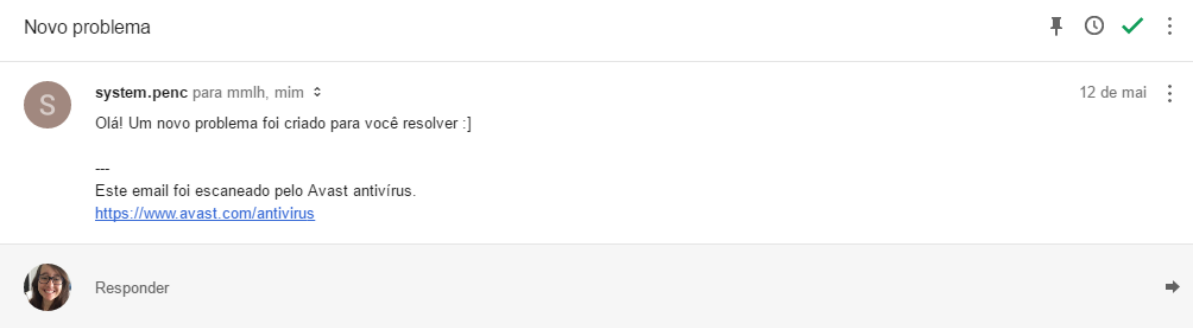


Figura 22: E-mail enviado pelo sistema penC

4.4 Curva de valor

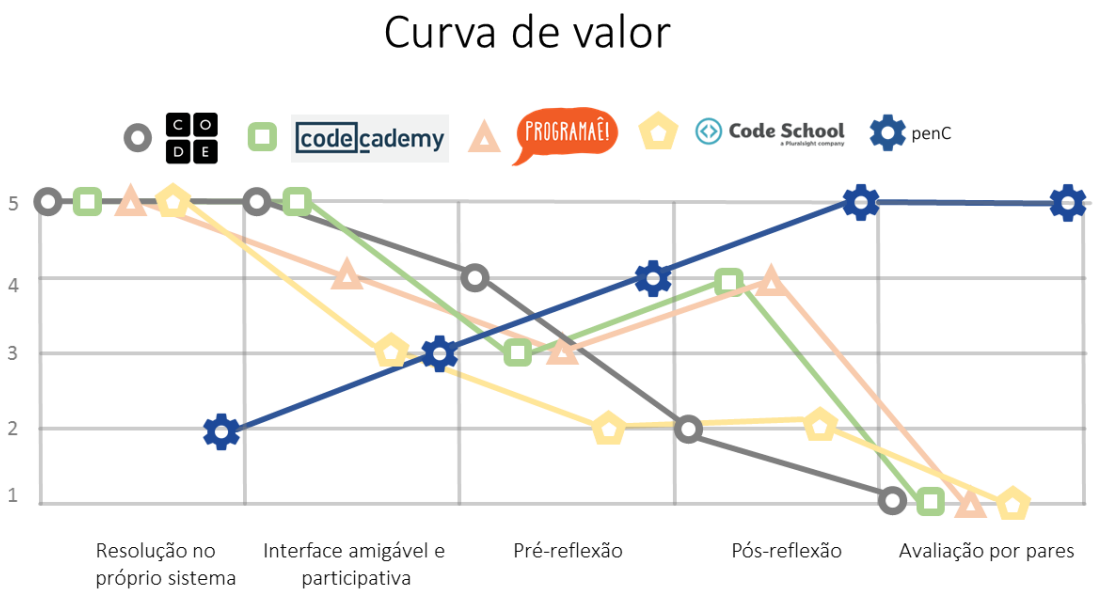


Figura 23: Curva de valor

Como pode ser observado na curva de valor acima, o penC se destaca de forma a se mostrar um sistema inovador em relação ao mercado, através dos critérios pós-reflexão e avaliação por pares que são partes essenciais na metodologia proposta em [SOARES, 2015] e que se mostraram ser de grande valia no experimento que será detalhado no capítulo seguinte. Para o próximo capítulo, também serão destacadas melhorias que foram analisadas no processo de desenvolvimento do sistema penC.

CAPÍTULO 5

Experimentação

Neste capítulo será explicado como foi realizado o experimento, e as conclusões que foram possíveis extrair com ele.

5.1 Visão geral

O experimento foi realizado com a turma da entrada de 2016.2 de Introdução à Programação do curso de Sistemas de Informação (SI), do Professor Fernando Castor, durante as aulas dos dias seis e oito de junho de 2016. Para a realização do experimento, foi requisitado ao *helpdesk* do Centro de Informática um servidor para hospedar o sistema e um servidor para hospedar o banco de dados. Desta maneira, os alunos poderiam responder aos problemas de sua casa através da VPN. Como o *helpdesk* não disponibilizou os servidores, o experimento foi feito com os alunos acessando o IP e uma porta específica de um computador que estava servindo de servidor para aquela rede. Ou seja, os alunos só puderam utilizar o sistema durante as aulas. Ao final do segundo dia, o experimento teve que ser interrompido porque o servidor da VPN parou de funcionar devido à quantidade de requisições feitas. Apesar das limitações impostas pela infraestrutura do Centro de Informática, o experimento foi muito proveitoso e muitas informações úteis e possíveis melhorias para o sistema puderam ser extraídas.

Na primeira aula foi explicado para os alunos o motivo do experimento e como é feita a contagem dos pontos na tela de *board* de resultado. Na segunda aula não houve necessidade de explicações. O experimento não possuía nenhum tipo de restrição, ou seja, os alunos poderiam discutir soluções entre si, poderiam pesquisar na internet e até analisar códigos passados. Não houve restrições, pois, era de intenção do experimento simular como se eles estivessem em casa estudando e não simular como se eles estivessem realizando uma prova. Apesar disto, o resultado era individual e no fim haveria apenas um ganhador: aquele que tivesse mais pontos no *board* de resultado.

Para os dois dias de experimento, foram disponibilizadas oito questões com níveis de dificuldades distintos para os alunos tentarem resolver durante a aula e foram criados dois questionários que os alunos deveriam responder antes e depois do experimento, afim de

entender o perfil do aluno, como ele se sentia em relação à programação e como ele se sentiu em relação ao sistema. Para o pré-experimento as seguintes perguntas foram feitas:

1. Qual o seu e-mail do CIn?

    Esta pergunta foi feita para realizar um comparativo individual do quanto a pessoa julgou saber de programação e o quanto ela demonstrou na realização dos problemas e como eram dois formulários distintos, era preciso identificar o aluno de alguma maneira.

2. Como você julga seu conhecimento em programação?

    Esta pergunta foi realizada com o intuito de saber o nível de proficiência em programação que o aluno se julga possuir.

3. O quanto você gosta de programação?

    Esta pergunta foi desenvolvida para entender se o quanto o aluno gosta de programação influencia na resolução das questões.

Afim de existir uma análise dos resultados comparando os resultados do pré-experimento com os do pós-experimento. Foram realizadas as seguintes perguntas para o pós-experimento:

1. Qual o seu e-mail do CIn?
2. Como você julga seu conhecimento em programação?
3. O quanto você gosta de programação?

    Estas três perguntas foram feitas novamente para poder entender se o sistema influenciaria nesses dois quesitos propostos (conhecimento em programação e o gosto por programação).

4. Para você, o quão fácil é utilizar o sistema?

    Esta pergunta foi feita para descobrir se o sistema está de fato fácil de ser utilizado pelos usuários.

5. O quanto você acha que o sistema contribuiu para o seu aprendizado?

    Este questionamento foi feito para entender se o sistema ajuda realmente os alunos aprenderem a programar.

6. Das funcionalidades do sistema, qual você mais gostou? Por quê?

    Esta pergunta foi desenvolvida para descobrir qual funcionalidade mais agradou os alunos e assim, gerar formas de melhor ainda mais essas funcionalidades.

7. Das funcionalidades do sistema, qual você menos gostou? Por quê?

Esta foi feita para identificar qual funcionalidade precisa ser revista de forma a facilitar a utilização do sistema pelo usuário.

8. Críticas, elogios ou sugestões?

Este foi um campo criado para os alunos poderem falar à vontade sobre o que acharam do sistema.

5.2 Perfil dos participantes

Neste experimento observou-se que os alunos não se acham tão bons em programação, já que a média de nota que eles dariam para si mesmos em termos de conhecimento foi de 5,333 levando em conta os vinte e sete alunos que participaram do experimento. Apesar disso é interessante observar que os alunos gostam de programação, pois a média da nota dada para o quanto o aluno gosta de programação foi de 8,8. Estas informações são dados constatados antes da realização do experimento.

5.3 Resultados da experimentação

Mesmo com o experimento sendo de apenas dois dias, puderam-se obter muitos dados acerca da influência do método proposto pelo sistema penC e também acerca das melhorias propostas. Como dito anteriormente, foram realizadas duas pesquisas (uma antes e uma depois do experimento) afim de entender como os alunos julgam seus conhecimentos em programação e o quanto eles gostam de programação tanto antes quanto após de eles utilizarem o sistema. Como pode ser observado no gráfico abaixo, houve um aumento da média dada pelos alunos tanto para o julgamento do conhecimento em programação, quanto para o quanto o aluno gosta de programação. Mesmo com o aumento aparentemente pequeno, se levarmos em consideração que foram apenas dois dias de experimento o ganho foi relativamente alto. Na figura 23 é possível visualizar o comparativo de médias das notas dadas pelos alunos do experimento, em relação ao julgamento do aluno acerca do seu conhecimento em programação e o quanto o aluno gosta de programação.

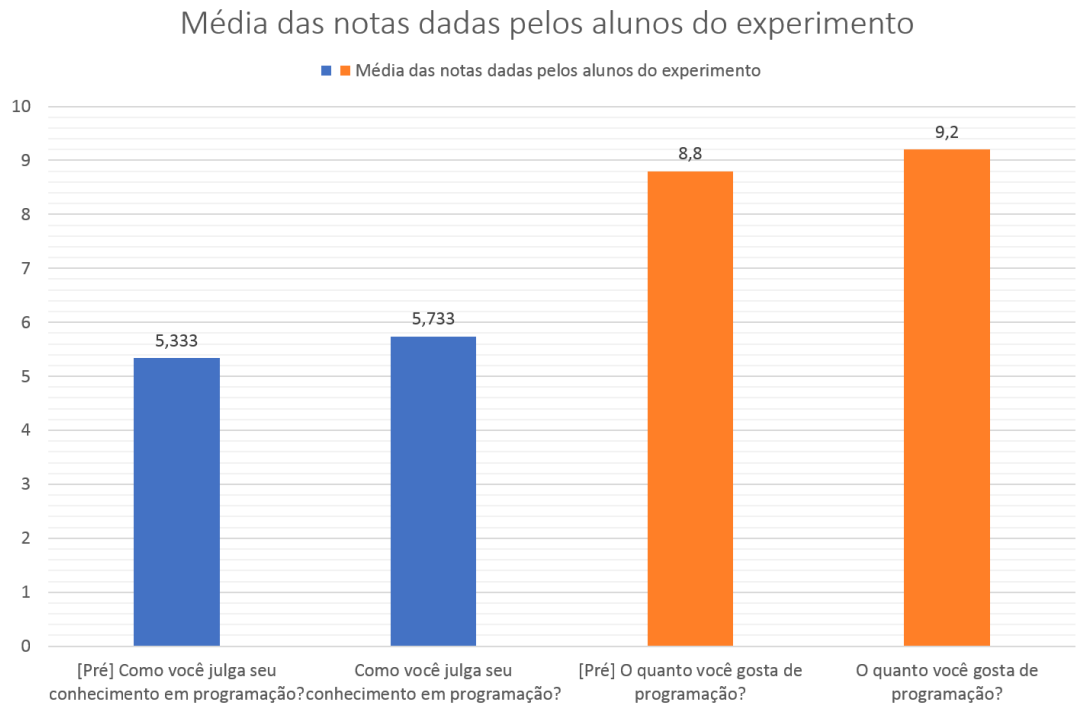


Figura 24: Gráfico da média das notas dadas pelos alunos do experimento

Explicando melhor o gráfico da figura 23, para a pergunta: "Como você julga seu conhecimento em programação?" A média formada pelas notas dos alunos antes do experimento foi de 5,333 e passou a ser 5,733 e para a pergunta: "O quanto você gosta de programação?" A média formada pelas notas dos alunos antes do experimento foi de 8,8 e se tornou 9,2. O experimento serviu como um meio de provar se o método proposto através do sistema penC de fato ajudava os alunos a aprenderem a programar, o que dentro do possível, foi comprovado.

Um outro ponto importante do experimento foi entender se o sistema estava fácil de ser utilizado e quais partes do sistema foram as mais importantes para os alunos. Com isto, poderia ser possível sugerir melhorias ao sistema penC de forma avaliada de que as melhorias propostas seriam realmente boas sugestões. Para medir esses dados, na pesquisa realizada pós experimento foi perguntado aos alunos o quão fácil era utilizar o sistema numa escala de 0 a 10. A média obtida foi de 8,07. Também foi feita uma pergunta para saber o quanto o aluno achou que o sistema contribuiu para o seu aprendizado e o resultado foi uma média de 7,6.

Perguntou-se também na pesquisa quais as funcionalidades que mais contribuíram para o aprendizado do aluno. 43% dos alunos informaram que foi a parte de avaliar as soluções de outros alunos, porque eles aprendiam outras formas de resolver o mesmo problema; 29% disseram que foi a parte de *gameficação* pois os estimulavam a ter mais

atenção nos problemas e mais vontade de tentar responder; 21% apontaram a parte de determinar quais eram os objetivos do problema porque eles passaram a entender melhor o motivo de estar fazendo aquela questão e não só o faziam de forma mecânica e os demais disseram que era a parte de refletir um problema pois poderiam analisar onde erraram, o que ajuda no aprendizado.

Foram propostas oito questões de diferentes níveis, para os alunos resolverem e elas trouxeram os seguintes resultados:

1. **Crie uma função recursiva para calcula n^x**

   Média de quão difícil os alunos acharam que seria fazer esta questão: 3,8/10

   Média de quanto os alunos acharam que conseguiriam fazer da questão: 8,13/10

2. **Escreva um procedimento recursivo Dec2Bin (n: integer) que dado um número decimal imprima a sua representação binária corretamente**

   Média de quão difícil os alunos acharam que seria fazer esta questão: 5,5/10

   Média de quanto os alunos acharam que conseguiriam fazer da questão: 8,3/10

3. **A função fatorial duplo é definida como o produto de todos os números naturais ímpares de 1 até algum número natural ímpar N. Assim, o fatorial duplo de 5 é 5!! = 1 * 3 * 5 = 15. Faça uma função recursiva que calcule o fatorial duplo.**

   Média de quão difícil os alunos acharam que seria fazer esta questão: 4,83/10

   Média de quanto os alunos acharam que conseguiriam fazer da questão: 8,29/10

4. **Faça uma função recursiva que permita inverter um número inteiro N. Ex: 123 - 321**

   Média de quão difícil os alunos acharam que seria fazer esta questão: 5,5/10

   Média de quanto os alunos acharam que conseguiriam fazer da questão: 8,44/10

5. **Escreva uma função recursiva que determine quantas vezes um dígito K ocorre em um número natural N. Por exemplo, o dígito 2 ocorre 3 vezes em 762021192**

   Média de quão difícil os alunos acharam que seria fazer esta questão: 6,14/10

   Média de quanto os alunos acharam que conseguiriam fazer da questão: 8,29/10

6. **A sequência de Padovan é uma sequência de naturais (n) definida pelos valores iniciais P(0) = P(1) = p(2) = 1 e a seguinte relação recursiva P(n) = P(n - 2) + P(n - 3) se n > 2 Alguns valores da sequência são: 1, 1, 1, 2, 2, 3, 4, 5, 7, 9, 12, 16, 21,**

28... Faça uma função recursiva que receba um número N e retorne o N-ésimo termo da sequência de Padovan

Média de quão difícil os alunos acharam que seria fazer esta questão: 3/10

Média de quanto os alunos acharam que conseguiriam fazer da questão: 8/10

7. **Uma criança está subindo as escadas com n degraus e ela pode subir de 1, 2 ou 3 degraus por vez. Implemente um método recursivo que conta quantas maneiras possíveis a criança pode subir as escadas**

   Nenhum aluno conseguiu fazer esta questão

8. **A index mágico em um array A[0 … n-1] é definido como sendo um index de forma que A[i] = i. Dado um array ordenado de inteiros, escreva um método recursivo que ache esse index mágico, se ele existe, em um array A. Faça esta questão utilizando busca binária**

   Nenhum aluno conseguiu fazer esta questão

Os problemas sete e oito, foram problemas mais difíceis propostos para testar o limite dos alunos. O problema sete exige um conhecimento matemático que os estudantes não lembravam/sabiam. Já o problema oito, exigia que o problema fosse resolvido por busca binária, que foi um assunto que eles não tinham visto em sala de aula ainda. Era esperado que os alunos pesquisassem para aprender para depois fazer a questão, todavia os estudantes informaram que não tiveram tempo suficiente para tal.

5.4 Proposta de melhorias do sistema

- Utilização do *guideline* do material design proposto pelo Google de forma a permitir ao usuário um design limpo, coeso, moderno e fácil de se utilizar. É possível acessar este *guideline* através do site https://material.google.com.
- Integrar *gameficação* no sistema para estimular os alunos a terem mais vontade de fazer as questões e utilizar tudo o que o sistema oferece. Na *gameficação* os alunos poderiam ganhar pontos por resolver uma questão; por realizar uma revisão; por fazer uma reflexão; por compartilhar um problema; por aceitar os objetivos das questões; por ganhar *badges*; por fazer todas as questões; por ser a primeira pessoa a terminar o problema; por tirar nota acima de sete em todas as questões.

  Para o aluno ganhar uma *badge*, o professor poderia definir requisitos como em quanto tempo os alunos levaram para terminar a questão; quantos dias antes de

terminar o prazo o aluno teria que terminar o problema e qual a nota mínima que o estudante tem que tirar na revisão.

Uma parte importante da sugestão da *gameficação* é permitir que os alunos observem quantos pontos eles e seus colegas de turma possuem e assim os estimular a fazerem as questões e a competição entre eles.

- Como complemento para a *gameficação*, os alunos poderiam visualizar os perfis dos seus companheiros de classe e as *badges* que cada colega possui.
- Possibilitar aos professores um ambiente em que eles possam compartilhar e discutir problemas através da ideia do REA. Para tal, eles poderiam ver os problemas que foram compartilhados por cada professor, os possibilitando de baixar e adaptar esses problemas para a realidade da sua turma. Seria interessante eles poderem criar grupos de discussões individuais ou em grupo, além de ser possível realizar um compartilhamento de arquivos dentro desses grupos de discussões.
- Deixar flexível, de forma que o professor escolha a quantidade de pessoas que irão revisar o problema em questão. Hoje cada aluno revisa apenas uma solução por problema, portanto pode-se gerar equívocos já que a correção também está sujeita a erros.
- Poderia existir um pequeno tutorial para cada parte do sistema de forma que sempre que a pessoa quisesse, ela poderia receber uma explicação detalhada de como funciona e como o usuário deve utilizar aquela parte específica do sistema.
- Poderia ser possível o professor habilitar ou não correção automática do problema. O professor colocaria casos de entradas e saídas para o problema e quando o aluno submetesse o seu código no sistema, ele automaticamente rodaria as entradas dadas e compararia com as saídas e informaria então para o aluno a porcentagem de acerto para aquela questão.
- Deixar destacado o código para a linguagem que a pessoa está utilizando como fazem os editores de texto Sublime Text e Notepad++.
- Ter a possibilidade de o aluno poder rodar o seu código diretamente no sistema, sem mais depender de um outro sistema para o aluno testar seu código antes de submeter no penC.
- Um espaço para os professores e os alunos disponibilizarem material de estudo como listas de exercícios, problemas resolvidos, slides das aulas, dentre outros.
- Na parte de revisar uma solução, o aluno revisor possuir a possibilidade de indicar exatamente onde estão possíveis erros no código.

- Permitir que o aluno configure o sistema para o notificar via e-mail quando o professor disponibilizar um novo problema, quando o aluno possuir uma nova solução para ele avaliar, etc...

CAPÍTULO 6

Conclusão

Neste capítulo serão detalhadas as contribuições deste trabalho e os trabalhos futuros possíveis.

6.1 Contribuições

O objetivo deste trabalho foi realizar o desenvolvimento do sistema penC, que é um sistema que traz os conceitos de autorregulação para ajudar os estudantes de computação a aprenderem o conteúdo, baseado no modelo proposto por [SOARES, 2015]. Outro objetivo era desenvolver tal sistema de forma que fosse respeitada a usabilidade, trazendo uma ótima experiência para aqueles que fossem se utilizar do sistema. E por fim, o último objetivo era propor melhorias para o sistema de forma que o mesmo se tornasse uma excelente plataforma de auxílio aos aspirantes e até estudantes avançados de computação. De forma que fosse possível também validar se o modelo proposto de aprendizado computacional aliado à autorregulação é de fato válido.

Após este estudo e desenvolvimento, foi possível desenvolver um sistema web que estava de acordo com o que foi proposto inicialmente como objetivo deste trabalho e foi possível também validar se o modelo proposto também era válido. Concluiu-se que o modelo proposto em [SOARES, 2015] de fato ajudava as pessoas aprenderam a programar e foi confirmado também que o sistema desenvolvido neste trabalho era fácil de ser utilizado e foi possível descrever uma série de melhorias para o modelo que foi utilizado como base para a criação do sistema.

As melhorias propostas para o sistema estão relacionadas tanto a uma usabilidade melhor para o usuário final (como por exemplo, utilizar o *guideline material design lite* da Google como padrão para a criação do design das telas do sistema), quanto na parte de novas funcionalidades do sistema que trariam benefícios para os estudantes e professores que viriam a utilizar o sistema. Tais melhorias serão descritas melhor na próxima seção.

6.2 Trabalhos futuros

Os resultados obtidos foram satisfatórios e suficientes para mostrar que o modelo penC funcionaria bem em contextos reais de utilização. Todavia, é recomendável a realização de trabalhos futuros para garantir resultados ainda melhores:

- Utilização do *guideline* do material design proposto pelo Google de forma a permitir ao usuário um design limpo, coeso, moderno e fácil de se utilizar.

- Integrar no sistema *gameficação* para estimular os alunos a terem mais vontade de fazer as questões e utilizar tudo o que o sistema oferece.

- Como complemento para a *gameficação*, os alunos poderiam visualizar os perfis dos seus companheiros de classe, juntamente com as badges.

- Possibilitar aos professores um ambiente no qual eles possam compartilhar e discutir problemas através da ideia do REA.

- Deixar a cargo do professor escolher a quantidade de pessoas que irão revisar o problema em questão. Hoje cada aluno revisa uma solução por problema, o que pode gerar equívocos, já que a correção também está sujeita a erros.

- Poderia existir um pequeno tutorial para cada parte do sistema de forma que sempre que a pessoa quisesse, ela poderia receber uma explicação detalhada de como o mesmo se comporta.

- Poderia ser possível para o professor habilitar ou não correção automática do problema. O professor a colocaria casos de entradas e saídas para o problema e quando o aluno submetesse o seu código no sistema, ele automaticamente rodaria as entradas dadas e compararia com as saídas e informaria então para o aluno a porcentagem de acertos para a questão.
- Deixar destacado o código para a linguagem que a pessoa está utilizando como fazem os editores de texto Sublime Text e Notepad++.

- Permitir ao aluno rodar o seu código diretamente no sistema, sem mais depender de um outro sistema para o aluno testar seu código antes de submeter no penC.

- Um espaço para os professores e os alunos disponibilizarem material de estudo como listas de exercícios, problemas resolvidos, slides das aulas, dentre outros.

- Na parte de revisar uma solução, permitir que o aluno revisor indique a possibilidade de indicar exatamente onde existem possíveis erros no código.

- Permitir que o aluno configure o sistema para o notificar via e-mail quando o professor disponibilizar um novo problema, quando o aluno possuir uma nova solução para ele avaliar, etc...

Referências Bibliográficas

ADLER; LUCENA; RUSSO; VIANNA; VIANN, "Design Thinking: Inovações nos Negócio. MJV Press, 2011.

AMIEL; MICHAEL; RICHARD, "Recursos Educacionais Abertos (REA): modelos para localização e adaptação." Educação Temática Digital 12: 112, 2011.

BENNEDSEN; CASPERSEN, Failure Rates in Introductory Programming, ACM SIGCSE Bulletin, Volume 39 Issue 2 (pp. 32-36), 2007.

BERGIN; REILLY; TRAYNOR, Examining the role of self-regulated learning on introductory programming performance. In: Proceedings of the 1st International Workshop on Computing Education Research. ACM. p. 81-86, 2005.

FRANÇA; ROZELMA; TEDESCO, "Um modelo para a aprendizagem do pensamento computacional aliado à autorregulação." Anais dos Workshops do Congresso Brasileiro de Informática na Educação. Vol. 4. No. 1, 2015.

GOOGLE, Exploring computational thinking. Acesso em: julho de 2016. Disponível em: <http://www.google.com/edu/computational-thinking>, 2013.

HADWIN; JÄRVELÄ; MILLER., Self-regulated, co-regulated, and socially shared regulation of learning. Handbook of self-regulation of learning and performance, v. 30, p. 65-84, 2011.

ISTE - International Society for Technology in Education; CSTA - Computer Science Teachers Association; NSF - National Science Foundation, Computational thinking: leadership toolkit. First Edition, 2011.

LU; FLETCHER, Thinking about computational thinking. In Proc. 40th Technical Symp. on Comp. Sci. Education, pages 260–264, New York, USA. ACM, 2009.

LUDOVICO; LUCA; GIUSEPPINA, Music Coding in Primary School. Smart Education and Smart e-Learning. Springer International Publishing. 449-458, 2015.

MATERIAL DESIGN.: Acesso em 22/03/2016. Disponível em: <http://www.google.com/design/spec/material-design/introduction.html#>, 2015.

MC FARLAND, The Rea;l Reason Why Everyone Should Learn to Code.: Acesso em abril de 2016. Disponível em: <http://blog.teamtreehouse.com/havent-started-programming-yet>, 2014.

MILLER, Why is learning to code so difficult? Acesso em abril de 2016. Disponível em: <https://www.quora.com/Why-is-learning-to-code-so-difficult>, 2014.

MITROVIC, Supporting self-explanation in a data normalizing tutor. In: Proceedings of the Workshop on Self-Regulated Learning in Educational Technologies, 2003.

TERENZINI; PATRICK; ERNEST; GREGORY, "Students' out-of-class experiences and their influence on learning and cognitive development: A literature review." Journal of college student development, 1996.

WING, Computational thinking. Commun. ACM, 49(3):33–35, 2006.

WING, Computational thinking and thinking about computing. Philos. Trans. R. Soc. A: Math. Phys. Eng. Sci. 366(1881), 2008.

WING, Research notebook: Computational thinking - What and Why? The Link. Spring, 2011.

YANG, "Creative VE activity using Value Curve." Samsung Electronics co., LTD.