KARL MARQUES SILVA
MATEUS FONSECA PELLUCHI

DYNAMIC CLASS - APLICATIVO PARA ENSINO DINÂMICO

São Paulo
2018

KARL MARQUES SILVA
MATEUS FONSECA PELLUCHI

DYNAMIC CLASS - APLICATIVO PARA ENSINO DINÂMICO

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Engenheiro Eletricista com ênfase em Computação.

São Paulo
2018

KARL MARQUES SILVA
MATEUS FONSECA PELLUCHI

DYNAMIC CLASS - APLICATIVO PARA ENSINO DINÂMICO

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Engenheiro Eletricista com ênfase em Computação.

Orientador:
Prof. Dr. Carlos Eduardo Cugnasca

São Paulo
2018

Um livro, uma caneta, uma criança e um professor podem mudar o mundo.

Malala Yousafzai

Eu nunca ensino aos meus alunos. Somente tento criar condições nas quais eles possam aprender.

Albert Einstein

RESUMO

Este trabalho propõe o desenvolvimento de um aplicativo móvel com o objetivo de auxiliar professores de matemática do ensino fundamental 2. Ele tem como justificativa o potencial a ser explorado na tecnologia da informação, especificamente m-learning, aplicada na área da educação presencial, quebrando o estereótipo de associar os smartphones à perda de foco e concentração por parte dos alunos, e utilizá-los como facilitadores de aprendizagem, e fornecer um recurso para que professores possam evitar discrepâncias de aprendizado entre alunos de uma mesma turma. O projeto foi elaborado como um aplicativo para Android, utilizando a Metodologia Agile, arquitetura MVC (Model-View-Controller) e persistência de dados com o banco de dados Google Firebase. O resultado é um *software* que permite o professor criar exercícios e aplicá-los para os alunos. Acertos e erros são contabilizados e uma análise de performance de um aluno ou classe é fornecida, possibilitando o professor adequar o nível de dificuldade dos exercícios propostos, criando um ensino personalizado.

Palavras-chave: Aplicativo, Ensino, Educação, Dinâmico, Dispositivos Móveis, M-learning, Mobile Learning

ABSTRACT

This work proposes the development of a mobile application with the aim to aid math elementary school teachers. Its reasoning is the potential to be explored on information technology, specially m-learning, applied in face-to-face education, breaking the stereotype that associates smartphones to loss of focus and concentration by students, making them a learning and engagement enhancer, providing a resource for teachers avoid learning discrepancies between class students. The project was made as a Android app, using Agile Methodology, MVC (Model-View-Controller) architecture and using Google Firebase database as data persistency. The result is a software that allows the teacher to create and apply exercises to students. Right and wrong answers are accounted and a performance analysis of a specific student, or the whole class, is provided, enabling the class teacher to adjust the difficulty level of exercises to each student, creating a personalized routine.

Keywords: App, Teaching, Education, Dynamic, Mobile Devices, M-learning, Mobile learning

LISTA DE FIGURAS

1 Comparação entre planejamento e realização das tarefas…… 24

2 Estrutura Analítica do Projeto - Aplicativo do aluno…….. 28

3 Estrutura Analítica do Projeto - Aplicativo do professor……..…. 29

4 Diagrama de classes do projeto……..... 30

5 Layout inicial de telas……..…... 31

6 Fluxograma de telas - Aplicativo do professor……... 32

7 Fluxograma de telas - Aplicativo do aluno…….. 32

8 Estrutura de banco de dados do projeto……..... 35

9 Estrutura de funcionamento do algoritmo…….. 38

LISTA DE SIGLAS

M-learning	Mobile learning
IOS	iPhone Operating System
RDS	Relational Database Service
EAP	Estrutura Analítica do Projeto
HH	Homem-Hora
RF	Requisito Funcional
RNF	Requisito Não Funcional
API	Application Program Interface
MVC	Model-View-Controller
MVVM	Model-View-ViewModel
XML	Extensible Markup Language
AES	Advanced Encryption Standard
UUID	Universally Unique Identifier
URL	Uniform Resource Locator

SUMÁRIO

1 Introdução 16

1.1 Objetivo 16

1.2 Justificativa 17

2 Aspectos Conceituais 18

2.1 Aplicações de TI no ensino..19

3 Tecnologias Utilizadas 21

4 Metodologia do Trabalho 22

4.1 Validação 24

5 Especificação de Requisitos do Sistema 26

5.1 Requisitos Funcionais e Não Funcionais 26

5.2 Estrutura Analítica do Projeto 28

5.3 Diagrama de Classes 30

5.4 Layout de Telas 31

6 Implementação 32

6.1 Arquitetura do Banco de Dados 33

6.2 Arquitetura MVC 36

6.3 Criptografia 37

6.4 Requisitos Funcionais e Não Funcionais - Resultados 38

7 Testes e Avaliação 41

7.1 Testes Modulares e Globais 41

7.2 Performance do Aplicativo e Firebase (Teste de Escalabilidade) 41

7.2.1 Teste de tempo máximo de resposta..41

7.2.2 Teste de quantidade de dados 42

7.3 Custo Financeiro de Manutenção ……..…… 42

8 Considerações Finais …… 44

8.1 Conclusões do Projeto de Formatura ……..44

8.2 Contribuições 44

8.3 Perspectivas de Continuidade.....45

8.3.1 Finalização do produto.....45

8.3.2 Expansão de plataformas..... 45

8.3.3 Incremento de funcionalidades.....46

Referências ……..... 47

1 INTRODUÇÃO

Este projeto tem como motivação principal explorar o potencial do uso da tecnologia na sala de aula, com foco no uso de dispositivos móveis por alunos e professores. Segundo o relatório “Um olhar sobre a educação”, publicado pela OCDE (2017), o Brasil é um dos países com menor investimento em educação básica, o que aumenta a importância e necessidade de soluções acessíveis e de baixo custo, que possam amenizar esse problema.

“A expressão "Tecnologia na Educação" deixa aberta a possibilidade de que tecnologias que tenham sido inventadas para finalidades totalmente alheias à educação, como é o caso do computador, possam, eventualmente, ficar tão ligadas a ela que se torna difícil imaginar como a educação era possível sem elas. A fala humana (conceitual), a escrita, e, mais recentemente, o livro impresso, também foram inventados, provavelmente, com propósitos menos nobres do que a educação em vista. Hoje, porém, a educação é quase inconcebível sem essas tecnologias. Segundo tudo indica, em poucos anos o computador em rede estará, com toda certeza, na mesma categoria. “ (CHAVES, 2004).

“Existem provas suficientes para afirmarmos que aplicações de M-learning (Mobile learning), se combinadas com pedagogia podem ser extremamente úteis para a educação. Até mesmo transformar o sistema educacional”. (DRIGAS; ANGELIDAKIS, 2017).

“Mobile learning possui inúmeros benefícios educacionais que facilitam o processo de ensino, e principalmente é barato, de fácil obtenção e possibilita o aprendizado em qualquer lugar, a qualquer momento”. (NAIL, 2017).

1.1 Objetivo

O objetivo do projeto é fornecer uma ferramenta *open source* e de baixo custo para o auxílio de professores.

O aplicativo permitirá educadores criarem classes virtuais, que serão acessadas pelos alunos cadastrados. Através do aplicativo, o professor acompanhará o desempenho dos alunos, que será salvo e poderá ser acessado, e poderá se sentir necessidade, adequar o nível dos exercícios, criando assim um ensino personalizado para cada aluno, evitando uma padronização pela média, que costuma ser aplicada, principalmente em salas com muitos alunos.

O projeto surge como uma ferramenta, que poderia ser utilizada pelos professores, e uma expansão nas oportunidades que o m-learning oferece para modernização e aumento de possibilidades de abordagens pedagógicas.

1.2 Justificativa

A educação é a chave para o desenvolvimento econômico e social de um país, são inúmeros os exemplos, como Coreia do Sul, Japão, Islândia; e num país extremamente desigual, como é o Brasil, as dificuldades educacionais enfrentadas para a parcela da população mais vulnerável economicamente, são ainda maiores. A expansão do m-learning é uma possibilidade para a diminuição da desigualdade no acesso à informação e educação.

Esse trabalho foca em dois grandes problemas enfrentados por educadores, principalmente em instituições públicas de ensino: o grande número de alunos, dificultando dar-lhes a devida atenção, e a padronização do ensino, que acaba desmotivando os alunos com dificuldade, pois não conseguem acompanhar, e também os alunos com facilidade na matéria, pois não são desafiados a aprofundar seus conhecimentos e não desenvolvem seu potencial.

A tecnologia que existe hoje é voltada principalmente para organização de horários e cronogramas, fóruns e mensagens entre alunos e professores e portais de ensino adaptativo com vídeos, substituindo o professor no ensino, com poucos softwares voltados a auxiliar o professor em aulas presenciais.

2 ASPECTOS CONCEITUAIS

O uso de tecnologia da informação, especialmente dispositivos móveis, vem crescendo nas escolas, e já é adotado de alguma maneira em 52% das instituições de ensino brasileiras (Cetic, 2016). Essa adoção tem sido devagar e conservadora, principalmente quando se compara com outras áreas da sociedade. Setores extremamente conservadores, como o bancário, já adotam em sua maioria soluções móveis.

Um grande empecilho para a propagação de aplicativos na educação é a crença de que dispositivos móveis atrapalham o ensino. O relatório da Unesco sobre *M-learning* (2012) reporta que há “uma atitude negativa da sociedade, que enxerga celulares como aparelhos que tiram a concentração”. Com a quebra deste tabu, soluções educativas em aplicativos móveis têm se tornado mais comuns.

Em uma pesquisa realizada com alunos de duas escolas similares, uma que permitia o uso de *smartphones*, e outra proibia (WALKER, 2013), mostrou que quando proibidos, os alunos utilizam muito menos os *smartphones* para a aprendizagem, mas o uso para lazer apresenta uma queda bem menos acentuada, mostrando que a utilização dos aparelhos em sala de aula, tende a ser benéfica.

Como literatura inicial, foi utilizado o livro “Ensino on-line e laboratório”, de Paulo Alves Garcia e José Sidnei Colombo Martini (GARCIA; MARTINI, 2007). Nele, foi apresentado o contexto atual de ensino, no qual foram levantadas oportunidades pouco exploradas da tecnologia disponível atual e sua atual no ensino.

Este trabalho pretende investir em algumas destas oportunidades, e com elas fornecer um meio para reforçar a sala de aula, e estendê-la para fora, sendo mais flexível e responsiva tanto ao aluno quanto professor.

Pontos relevantes foram levantados, como sobre o foco em fornecer uma ferramenta para auxiliar o professor, e não substituí-lo. Certos aspectos de aprendizado são inerentes à interação entre aluno e professor, e foi concluído que qualquer tentativa de substituição dessa interação por *software* seria infrutífero. Ao invés disso, esse trabalho pretende fornecer mais informações ao professor e aluno por meio do acompanhamento contínuo digitalmente.

Também foi levantada a preocupação quanto ao aproveitamento do aluno, e uma metodologia para a avaliação e acompanhamento do aluno para melhorar a

qualidade do ensino. Tal metodologia poderá ser utilizada como base para nosso algoritmo.

2.1 Aplicações de TI no ensino

O uso de tecnologia da informação para auxiliar no ensino se deu de maneira atrasada quando comparada a outras áreas entretenimento e comunicação. No relatório “Mapeamento Edtech” (CIEB, 2018), é evidenciado que a utilização da computação e da internet na educação ganhou força principalmente com plataformas de cursos online como Udemy e Khan academy, e plataformas de aulas online, voltadas para o preparo para o vestibular e com ensino adaptativo, como Stoodi e Descomplica.

Analisando o “Mapeamento Edtech”, nota-se que *softwares* com o objetivo de auxiliar o professor são menos comuns. Em sua maioria são utilizados *softwares* específicos de uma instituição, ou um *software* não especificamente voltado para o ensino, como ferramentas Google, Dropbox ou Whatsapp. São *softwares* que facilitam comunicação e entrega de tarefas, mas não aumentam as possibilidades do professor.

No mercado mobile, existem 2 ferramentas, com integração web, que servem para auxiliar professores no desenvolvimento de aulas, e gerar informação que possa ser utilizada para direcionar os próximos passos. O Google Classroom, e o TeacherKit (TEACHERKIT, 2018).

TeacherKit é um *software* que foi desenvolvido em 2014 por um grupo de professores que tinham a necessidade de organizar as informações de seus alunos e do desempenho deles em um lugar só, com possibilidade de explorar esses dados com gráficos e tabelas. Esse *software* não é utilizado para aplicação de exercícios. O professor insere o desempenho do aluno manualmente, além de outros dados, como presença, participação, comportamento e o *software* fornece relatórios baseados nesses dados(TEACHERKIT, 2018).

O Google Classroom é mais semelhante com a proposta deste trabalho. Ele é uma ferramenta web, com aplicativo mobile, que permite a criação de classes, divisão por capítulos e aplicação de tarefas, correção e atribuição de notas para essas tarefas e visualizar as notas dos alunos. Também proporciona integração com

as ferramentas Google, facilitando avisos e divulgação de conteúdo (GOOGLE, 2018).

Tanto TeacherKit quanto Google Classroom não suprem as necessidades que esse projeto se propõe a suprir. Através do TeacherKit não é possível aplicar exercícios e o Google Classroom é pouco dinâmico para aplicar exercícios, sendo focado para tarefas maiores como entrega de trabalhos por parte dos alunos. Também é preciso criar a tarefa e atribuí-la a um grupo de alunos, não há a possibilidade de popular um banco de dados e oferecer rotinas de exercícios personalizadas para alunos, utilizando esse banco populado.

Os sistemas disponíveis na área de *softwares* auxiliares na educação mostra que esse campo ainda é pouco explorado, e corrobora com a importância, e até certo ponto ineditismo do projeto aqui proposto.

3 TECNOLOGIAS UTILIZADAS

Este trabalho desenvolveu um aplicativo para dispositivos móveis. A escolha deste formato de *software* é devido à praticidade e a possibilidade da utilização em sala de aula.

Os dois principais sistemas operacionais utilizados em dispositivos móveis atualmente são o Android e iOS (iPhone Operating System), possuindo juntos mais de 99% do mercado em 2017 (GARTNER, 2018). O desenvolvimento para iOS exige hardware específico da fabricante Apple, ou a utilização de um ambiente de desenvolvimento que permita a exportação para aparelhos iOS. Foram analisadas duas possibilidades de desenvolvimento para ambas as plataformas, Unity (UNITY, 2018) e Xamarin (MICROSOFT, 2018). Por ser voltada para jogos, Unity não seria a tecnologia ideal para o desenvolvimento desse projeto, e Xamarin, mostrou-se um bom ambiente para desenvolvimento para dispositivos móveis, porém, seria um fator limitante para a escolha de outras tecnologias e possui uma documentação muito mais escassa do que a de Android Studio. Por esses motivos, foi decidido que o desenvolvimento seria exclusivamente feito para Android, utilizando seu SDK (kit de desenvolvimento de *software*).

Para a persistência do aplicativo, foi decidido que seria utilizado um sistema de banco de dados. Priorizando a facilidade de implementação e a integração com desenvolvimento Android, a solidez do sistema e a documentação acessível, destacaram-se os banco de dados da Amazon AWS (AMAZON, 2018), o RDS (*Relational Database Service*) (relacional), o DynamoDB (não relacional) e o da Google, o Firebase Realtime Database (não relacional). A utilização dos serviços da Amazon possibilitaria o uso de dois tipos de banco de dados, além da integração de diversos serviços da Amazon, como o Amazon S3 (serviço de armazenamento) e o Amazon Neptune (serviço de banco de dados gráfico). Já o Firebase, possui a vantagem de ser desenvolvido pela Google, mesma desenvolvedora do Android, o que facilita ainda mais sua implementação, e por ser um banco de dados voltado para aplicações em tempo real. Optando por desempenho e simplicidade, o Firebase foi escolhido.

4 METODOLOGIA DO TRABALHO

Para a elaboração do projeto, foi utilizada a Metodologia Agile de desenvolvimento. Essa metodologia já foi amplamente aplicada para desenvolvimento de *software*, e continua como uma das principais metodologias de desenvolvimento no mercado. A Metodologia Agile é eficiente para projetos com entregáveis curtos e incrementais, tornando-a ideal para esse trabalho.

As histórias serão baseadas nos requisitos levantados para o projeto, durante a fase de especificação, e a pontuação inicial e resultados dos primeiros *sprints* no forneceram uma estimativa do tempo de desenvolvimento do projeto. Como o grupo não possuía experiência no desenvolvimento de aplicativos para o sistema operacional Android, estimativas de tempo antes da conclusão não seriam confiáveis.

A partir destes dados, foi estimado o restante do cronograma para o projeto, usando como base nossa pontuação para cada história.

Para elaboração dos entregáveis, dividiu-se o *software* através do método EAP (Estrutura Analítica do Projeto), obtendo tarefas curtas e concisas.

O Scrum foi coordenado por meio da plataforma Trello. É uma ferramenta de simples utilização, e que cumpre bem as necessidades do Scrum para um grupo pequeno. Outras ferramentas foram consideradas, mas pela simplicidade, o Trello foi utilizado.

Junto com o Scrum, foi utilizado o conceito de código modular e reutilizável. A solução de separação de aplicativo para “cliente” e “prestador de serviço” é a estratégia mais comum no mercado, pois evita código inutilizado no aplicativo final, e permite um desenvolvimento de código simples e modular, sem tratamento excessivo de condicionais.

Os *sprints* foram divididos entre as funcionalidades do sistema como um todo, englobando ambos os aplicativos. O sistema foi dividido de uma forma que fosse possível no final de cada *sprint*, obter-se um *software* pronto, com todas as funcionalidades testadas adequadamente, e a *sprint* seguinte incrementaria esse *software* com uma nova funcionalidade.

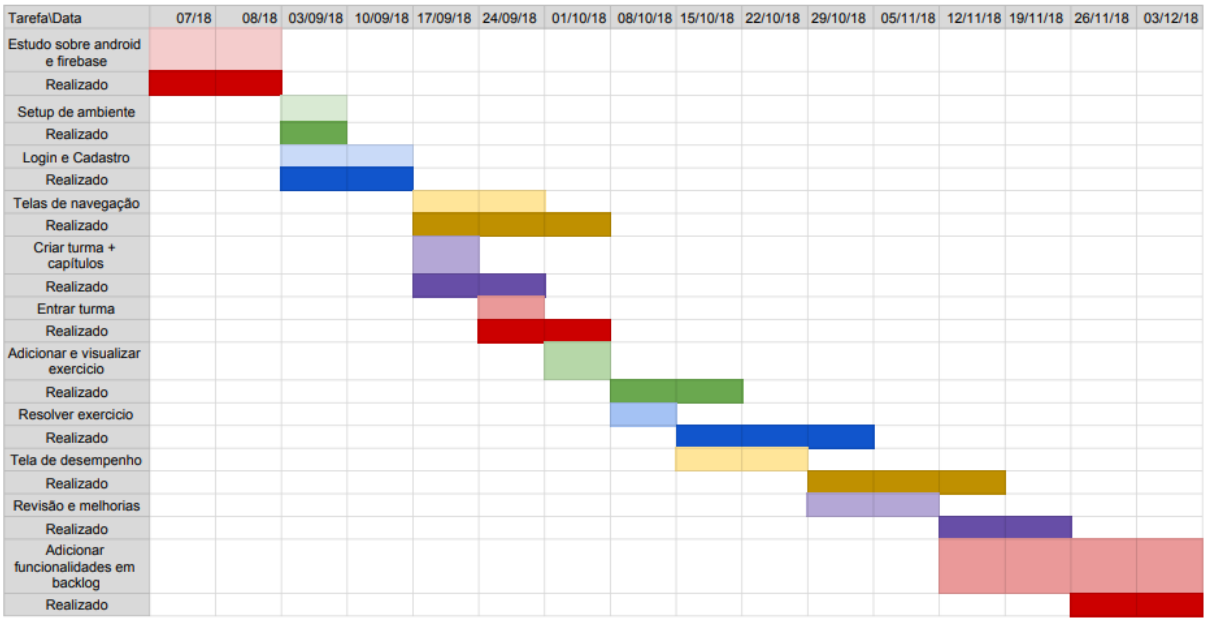
Seguindo esse critério, foram definidas cinco *sprints*:

- **Login e Cadastro:** a primeira *sprint* iniciou o projeto, criando o repositório e o banco de dados Firebase e implementou métodos de *login* e cadastro para o usuário.
- **Criação e participação de turmas:** essa *sprint* deverá implementou a funcionalidade de criação de turmas e capítulos através do aplicativo do professor, e a função de ingressar turma, através do aplicativo do aluno, implementando também as telas necessárias para esse fluxo.
- **Criação e resolução de exercícios:** essa *sprint* deverá implementar uma rotina de criação de exercício para o professor, respeitando os requisitos funcionais do sistema, o método de aplicação dos exercícios para os alunos e implementar a rotina de resolução de exercício, salvando o desempenho do aluno no lugar adequado do banco de dados.
- **Análise de desempenho:** essa *sprint* deverá implementou a tela de desempenho, a ser acessada pelo professor. Essa tela tem os dados de desempenho da turma, de alunos individualmente, para um capítulo específico ou para a matéria inteira que o professor leciona.
- **Revisão, melhorias e backlog:** essa *sprint* se repetiu até o fim do prazo de conclusão deste trabalho. Nela o sistema foi revisado, aprimorado onde possível, e funções não essenciais, mas que poderiam melhorar a sua utilização ou desempenho foram implementadas.

Após a definição das *sprints*, foi definido como média de trabalho 16 HH (Homem-Hora) por semana, levando em conta a disponibilidade dos dois integrantes e a variância ao longo das semanas, principalmente por motivos acadêmicos. As sprints foram divididas em tarefas estimadas em até duas semanas de duração, com as últimas quatro semanas dedicadas apenas a melhorias, revisão e tarefas em *backlog*. A tarefa de backlog representa 40% do tempo das tarefas anteriores. Essa medida foi tomada é comum o atraso na entrega de projetos, devido a estimativas otimistas (GARCIA, 2013).

A Figura 1 apresenta a comparação entre as tarefas planejadas e as realizadas:

Figura 1: Comparação entre planejamento e realização das tarefas



4.1 Validação

A validação do projeto foi feita por meio de testes de funcionalidade do programa. Esses testes foram realizados ao final de cada *sprint*, e serão obrigatórios para a sua finalização.Os testes foram divididos em três categorias:

- **Teste de integridade do aplicativo**: foi checado se a interface gráfica e a navegação no aplicativo se comportou de maneira adequada. Neste teste botões, imagens, textos e dados exibidos foram checados quanto à realização de funções e disposição na tela.

- **Teste de integridade dos dados**: foi checado se os dados foram corretamente acessados e enviados pelo programa aplicativo para o banco de dados. Nesse teste, o programa aplicativo foi usado para o envio de diversos dados (como a submissão de exercícios, rotinas, e soluções, cadastros de novos usuários, dentre outros), e também foram criados bancos de dado de testes, que serviram como simulação para diferentes casos de uso (criação de turma, resolução de exercício, entre outros).

- **Teste de funcionalidade**: foi checado se as funcionalidades do aplicativo funcionaram corretamente. Esse teste avaliou a integridade dos testes anteriores, realizando o fluxo completo das funcionalidades da *sprint* testada, validando-a ou não, para a próxima *sprint*.

5 ESPECIFICAÇÃO DE REQUISITOS DO SISTEMA

O sistema possui dois atores: Alunos e Professores.

Um professor consegue criar turmas, que são compostas por capítulos, e os capítulos possuem exercícios. Os exercícios possuem nível de dificuldade e são salvos no banco de dados. Os exercícios podem conter imagem e/ou texto, e possuem uma resposta que pode ser exata, múltipla escolha, ou um mínimo e máximo aceito.

Alunos podem ingressar classes, através do Id e senha desta, caso seja protegida. Uma vez na classe, o aluno consegue resolver os exercícios propostos pelo professor, acessar os capítulos e o perfil do professor responsável. Os exercícios disponíveis para o aluno poderão ser controlados pelo professor responsável, coordenando o nível de dificuldade individualmente, personalizando o ensino.

O professor tem acesso ao rendimento dos alunos em classes que administra, através da tela de performance de alunos, podendo acessar o desempenho em um capítulo específico ou em todos, de um aluno específico, ou a média de todos os alunos. Munido desta informação, o professor pode readequar os exercícios oferecidos a um aluno, se assim achar necessário, conseguindo assim estimular aquele com facilidade e reforçar o ensino daquele que está com dificuldades.

Serão desenvolvidos dois aplicativos, similares, um voltado ao professor, outro voltado ao aluno.

5.1 Requisitos Funcionais e Não Funcionais

RF01: O usuário deverá conseguir se cadastrar através do aplicativo.
RF01.1: O usuário deverá conseguir mudar sua senha pelo aplicativo.
RF02: O professor cadastrado deverá conseguir criar turmas através do aplicativo.
RF03: O aluno cadastrado deverá conseguir ingressar em turmas através do aplicativo.
RF04: O professor poderá adicionar exercícios aos capítulos das turmas.
RF04.1: O professor poderá adicionar exercícios com horário agendado.

RF04.2: O exercício poderá ter dificuldade, dada pelo professor.
RF05: O desempenho do aluno poderá ser visto pelo professor da turma e pelo próprio aluno.
RF06: O aluno deverá conseguir responder os exercícios propostos através do aplicativo.
RF06.1: O aluno deverá receber do aplicativo, uma resposta de acerto ou erro no exercício resolvido
RF07: O professor poderá criar programas de progressão em um capítulo, baseado no número de exercícios a serem resolvidos de cada dificuldade.

RNF01: Os aplicativos do professor e do aluno serão separados
RNF02: A turma poderá ter uma senha de ingresso definida pelo professor.
RNF03: O Aplicativo será desenvolvido para a plataforma Android
RNF04: O aplicativo se comunicará com o banco de dados não-relacional Firebase
RNF05: O aplicativo não deverá exceder o tempo de resposta de 1 minuto, quando conectado à internet
RNF06: Os exercícios podem ser imagens ou textos

5.2 Estrutura Analítica do Projeto:

A seguir são apresentadas as EAPs do projeto, para os atores Aluno e Professor, nas Figuras 2 e 3.

Figura 2: Estrutura Analítica do Projeto - Aplicativo do aluno

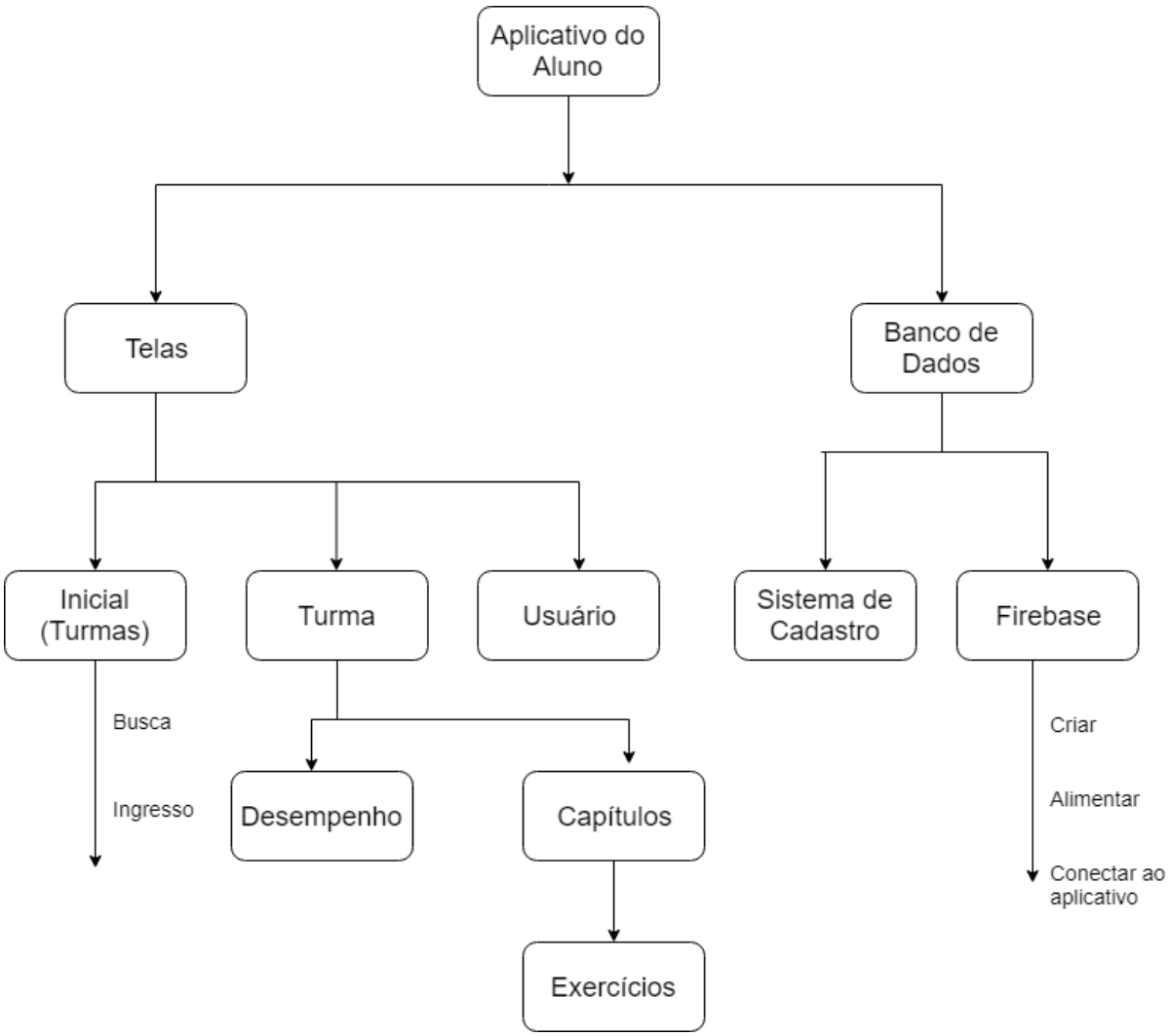
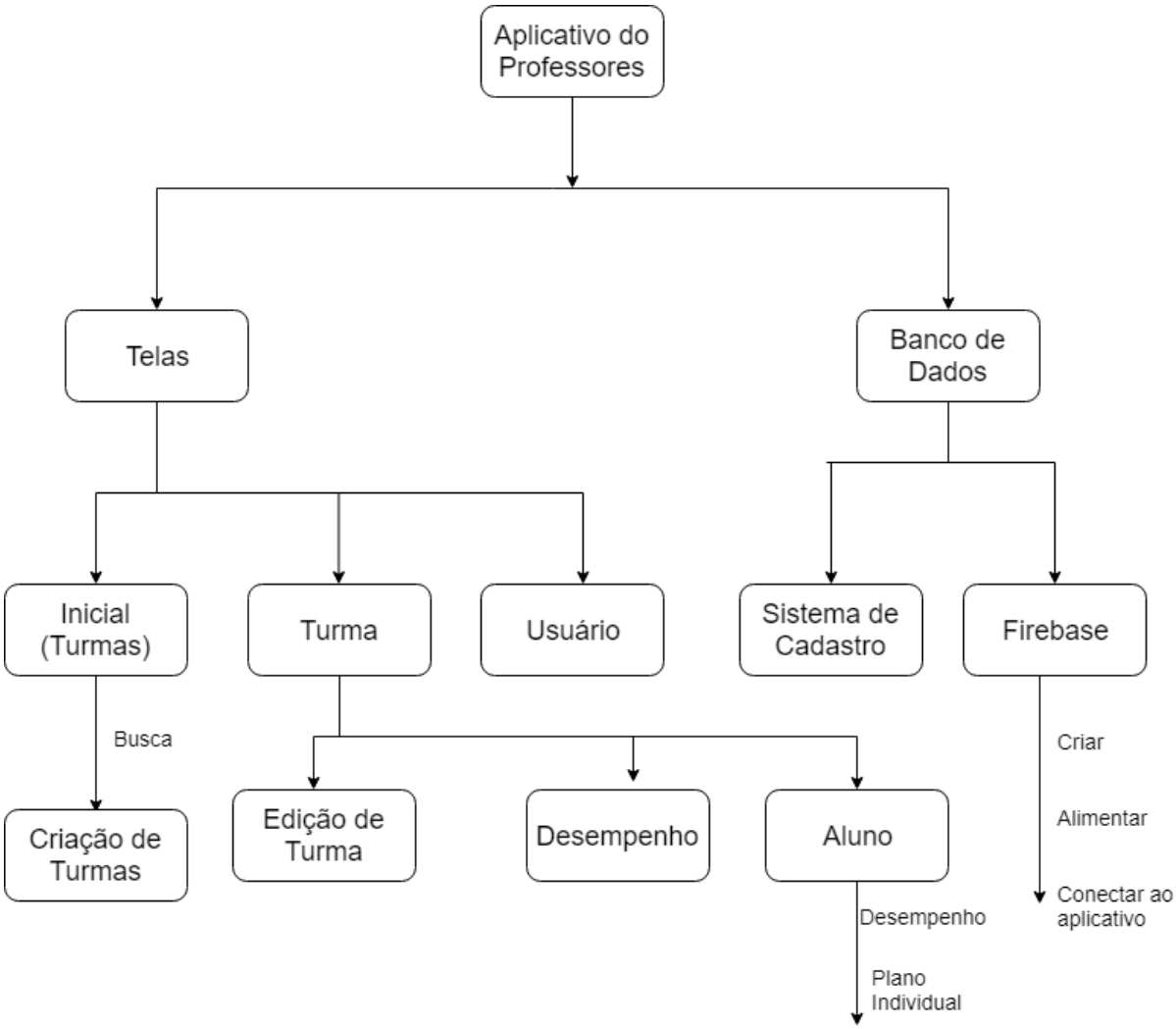


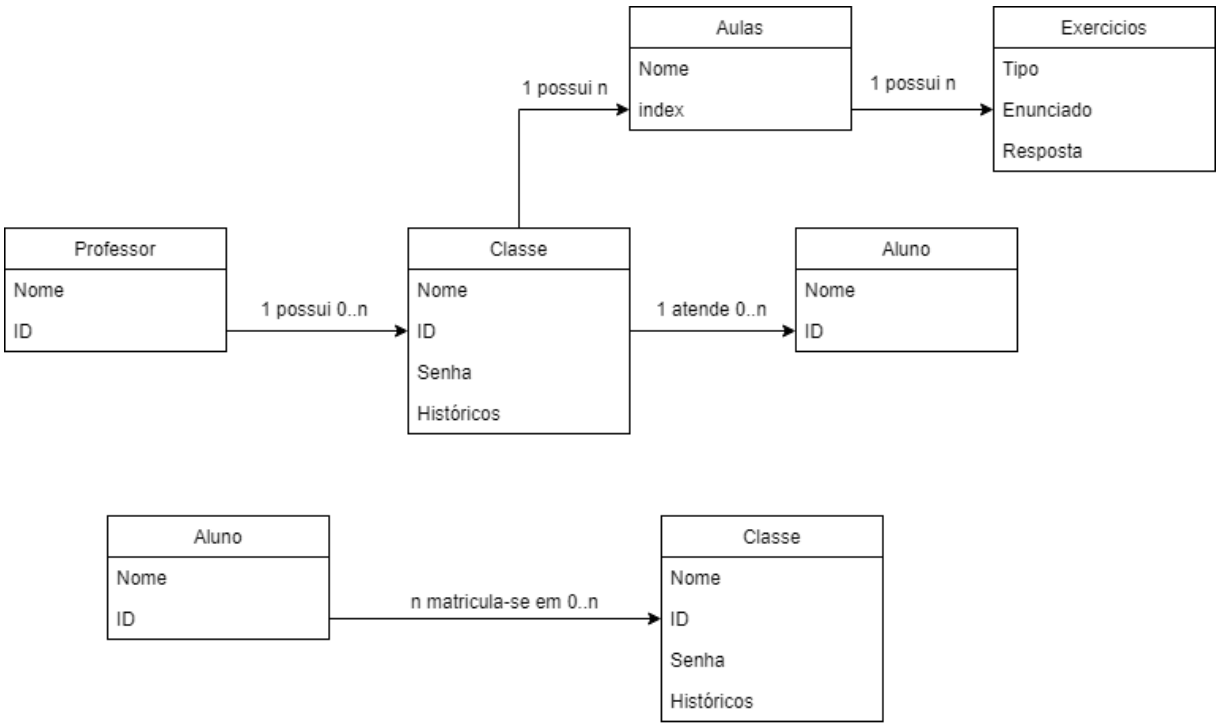
Figura 3: Estrutura Analítica do Projeto - Aplicativo do professor



5.3 Diagrama de Classes

A Figura 4 apresenta o diagrama de classes utilizado para a elaboração do projeto:

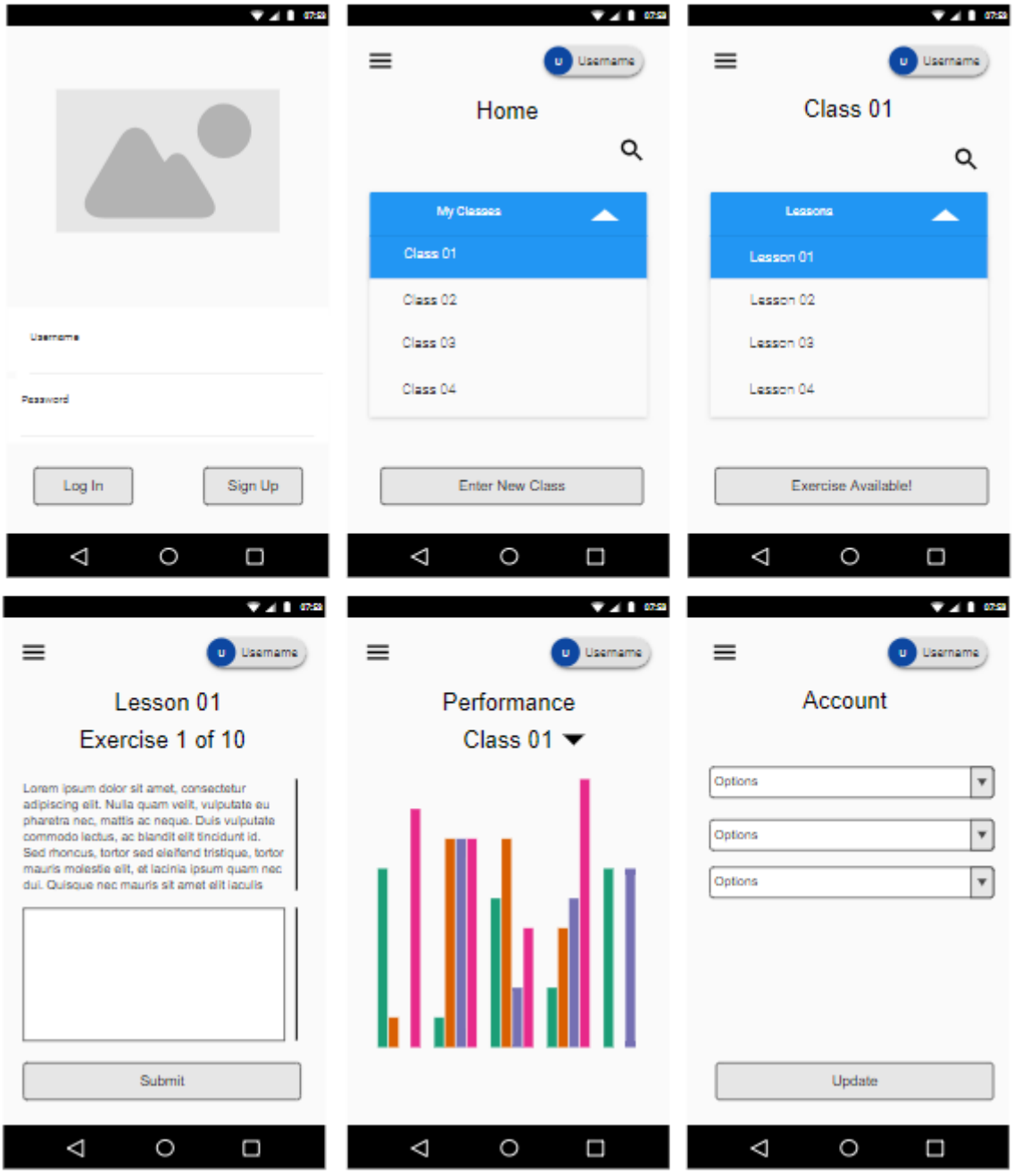
Figura 4: Diagrama de classes do projeto



5.4 Layout de Telas

A Figura 5 apresenta os *layouts* iniciais do projeto, contendo suas telas principais. Mais telas foram adicionadas em *sprints* seguintes, usando como base estes *layouts*.

Figura 5: *Layout* inicial de telas



6 IMPLEMENTAÇÃO

A implementação foi conduzida de maneira incremental, seguindo as *sprints* definidas pela Metodologia Agile. Foram desenvolvidas dez telas para o aplicativo do aluno e quinze telas para o aplicativo do professor. Também foram desenvolvidas classes auxiliares para a comunicação com o banco de dados, e um módulo de criptografia. Também foi criado e estruturado um banco de dados não relacional no Firebase e uma instância do Firebase Storage para salvar as imagens. Os aplicativos do professor e alunos seguem os fluxogramas apresentados nas Figuras 6 e 7.

Figura 6: Fluxograma de telas - Aplicativo do professor

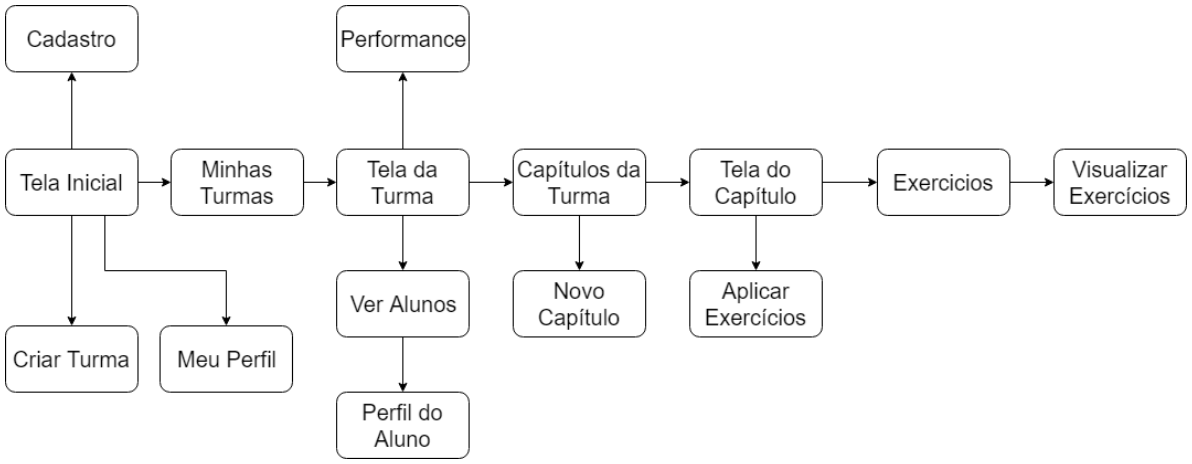
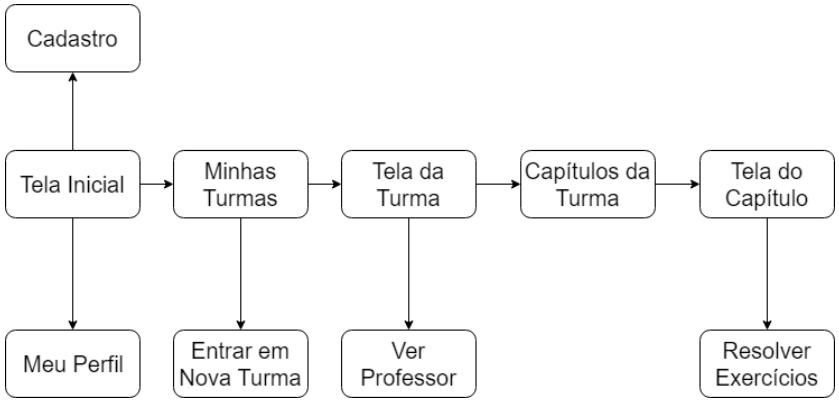


Figura 7: Fluxograma de telas - Aplicativo do aluno



6.1 Arquitetura do Banco de Dados

Para a estruturação do banco de dados, o Firebase da Google foi escolhido, por ser de fácil integração com o Android, e ser do tipo não relacional, o que facilita a implementação de algumas funcionalidades do projeto, e torna o banco mais adequado a mudanças imprevistas ao longo do projeto.

Também possui ferramentas nativas, que se adequam ao desenvolvimento deste trabalho, como o sistema já implementado de cadastro e login, o Cloud Storage, sistema de armazenamento de arquivos, utilizado para armazenar imagens dos exercícios criados, e o Firebase Analytics, plataforma com dados sobre o uso do aplicativo, como retenção de usuário, acessos diários e volume de dados salvos e baixados, que permite analisar a escalabilidade do projeto.

Por ser um banco de dados não relacional, o projeto demanda uma estruturação de dados mais cuidadosa, pois se torna mais suscetível a erros e à má utilização dos recursos do usuário, principalmente referente ao volume de dados baixados.

O Firebase é criado como um arquivo do formato Json na nuvem, e à partir da raíz do banco de dados, pode-se criar nós, que possuem um valor definido, ou nós filhos associados a ele. Para acessar informações, define-se o caminho para o nó desejado, e então cria-se um “*DataSnapshot*”, objeto nativo da API (*Application Program Interface*) do Firebase, que é uma cópia do nó especificado e todos os seus nós filhos. Um problema enfrentado por programadores que utilizam o Firebase é a falta de escalabilidade quando a arquitetura do banco de dados não é pensada de maneira cuidadosa, pois muitas vezes o usuário necessita baixar um alto volume de dados, para obter um baixo volume de informação.

Uma maneira de suplantar esse problema é a duplicação de dados. Dados muito acessados, e que estão sob um nó muito populado, preferencialmente devem ser duplicados, em um local que possam ser acessados sem a necessidade de baixar um alto volume de dados não requeridos. Essa solução se torna um paradigma, em que o programador deve construir o banco de dados mensurando a organização e volume de dados salvos, contra a escalabilidade e volume de dados baixado pelo usuário.

Neste projeto, o banco de dados foi inicialmente estruturado com três nós filhos diretos da raíz: “Teachers”, com informações sobre os professores e também o código e nome das turmas que lecionam, “Students”, com informações sobre os estudantes e também o código e nome das turmas que participam, e “Classes”, que são as informações da turma, junto com capítulos e exercícios. Posteriormente, através da leitura dos Analytics, foi detectado um ponto de melhorias na rotina de ingresso de turma, feita pelo aluno, por ter a necessidade de baixar o nó “Classes” inteiro para encontrar a turma através do código. A informação de nome e senha foi duplicada em um novo nó “*ClassesPassword*”, diminuindo o desperdício da dados do usuário. No caso de detecção de volume de dados superior a 2 Mb, seria benéfico ao projeto uma nova divisão, salvando as informações sobre a classe em um outro nó, evitando a necessidade de o usuário baixar a classe inteira. Nesse ponto do projeto, não se mostrou necessária essa mudança.

A Figura 8 apresenta a estrutura de banco de dados do projeto a seguir:

Figura 8: Estrutura de banco de dados do projeto

6.2 Arquitetura MVC

No início do desenvolvimento do projeto, dois tipos de arquitetura foram consideradas para o programa aplicativo: a Model-VIew-Controller (MVC) e Model-View-ViewModel (MVVM). A arquitetura MVC é uma das implementadas para o desenvolvimento de aplicativos para dispositivos móveis. Já a MVVM surgiu com base na MVC, resolvendo um problema enfrentado em projetos com poucas telas e muito conteúdo, que era um *Controller* excessivamente grande, dificultando testes e compreensão do código. A arquitetura MVC foi escolhida por tratar-se de um projeto com muitas telas e pouca necessidade de controle explícito por parte do *Controller*, e por MVC ser uma arquitetura mais disseminada e bem estabelecida.

O modelo MVC é a divisão do *software* entre o *Model*, responsável pelos dados e pelo estado do projeto e lógica do negócio, a *View*, que é a parte gráfica e visual do projeto, e a interação com o usuário, e o *Controller*, responsável pela lógica interna do projeto, recebe as entradas da *View* e administra as informações recebidas do *Model*. O grande benefício dessa arquitetura é a modularidade, auxiliando no desenvolvimento de um código desacoplado, facilitando a aplicação de testes unitários.

No desenvolvimento para Android já há uma divisão bem determinada da View, que são arquivos no formato XML (*Extensible Markup Language*), separados do código em Java. As *Activities*, classes em Java já criadas pelo Android para o controle de telas, foram definidas como os controllers. Já o *Model*, foi definido como o Firebase, para a maioria das telas, auxiliado em duas telas de adapters, classes Java criadas como intermediárias entre o Firebase e a *Activity*, responsável pela aquisição e organização dos dados, e métodos de obtenção dos dados para a *Activity*.

O modelo MVC foi seguido de maneira menos rigorosas nas telas mais simples, já que seu benefício não seria tão grande. Já nas telas mais complexas, de aplicação de exercícios e a tela de verificação de performance dos alunos, a complexidade exigiu um maior rigor e foi observado um significante auxílio no desenvolvimento e nos testes, possibilitando inclusive o desenvolvimento do *Model* por um integrante, o desenvolvimento posterior da *View* e do *Controller* pelo segundo integrante.

Foi constatado através do projeto que a utilização da arquitetura MVC no desenvolvimento do aplicativo móvel é extremamente útil para a estruturação, organização e teste do sistema a ser desenvolvido.

6.3 Criptografia

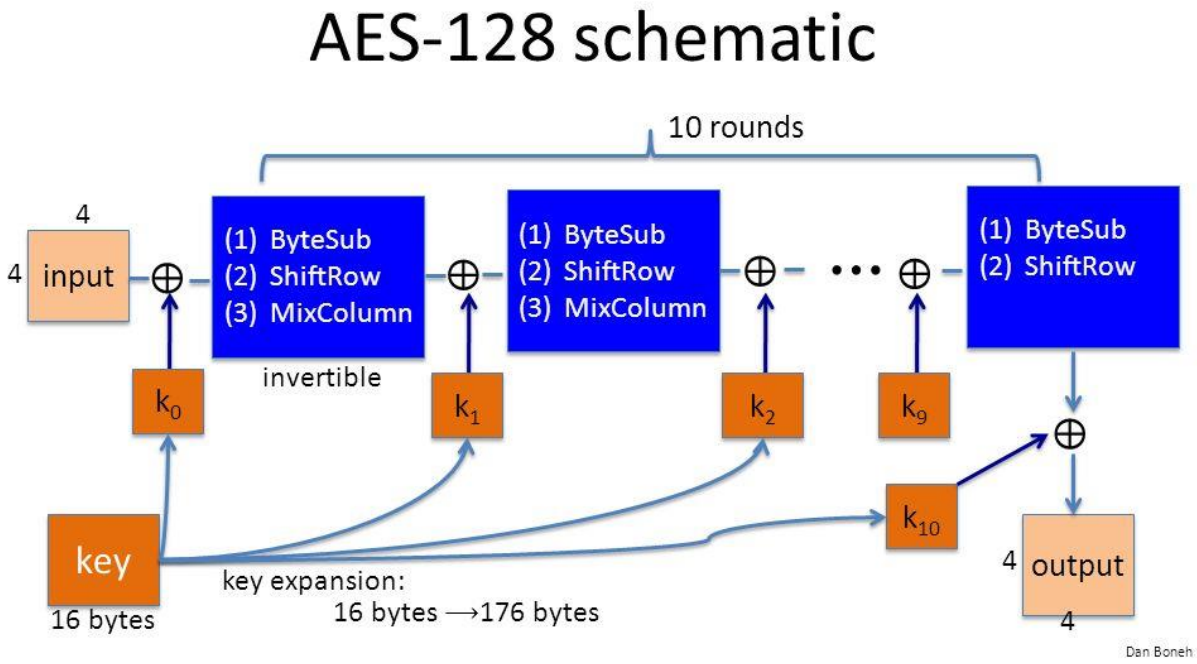
Um requisito levantado ao longo da implementação do projeto, foi a necessidade da proteção de dados sensíveis dos usuários. A tecnologia adotada para o banco de dados é de domínio da Google, sendo levantado que a exposição de dados dos professores, e principalmente de alunos, poderia ser um obstáculo para a adoção do aplicativo, pois os dados estariam vulneráveis para *data mining*. A solução para este problema foi a incorporação de uma camada de criptografia sobre os dados sensíveis dos usuários, protegendo-os de uma possível utilização por terceiros.

Por ser um programa aplicativo que utiliza comunicação assíncrona, e alunos devem ter acesso ao contato de seus professores, e os professores devem ter acesso ao contato de seus alunos, foi decidido que o tipo de criptografia mais adequada para o projeto seria a criptografia de chave simétrica. Assim, qualquer usuário com acesso à chave da criptografia consegue ler os dados.

O algoritmo escolhido para a criptografia foi o *Advanced Encryption Standard* (AES). Trata-se de um algoritmo de chave simétrica extremamente robusto, e de processamento rápido, tornando-o indicado para aplicações móveis.

A Figura 9 apresenta a estrutura de funcionamento do algoritmo AES.

Figura 9: Estrutura de funcionamento do algoritmo AES



Fonte: crypto.stanford.edu

Como chave de criptografia foi utilizado o *Universally Unique Identifier* (UUID), fornecido pelo Firebase, adicionando-se quatro outros caracteres fixos. O UUID trata-se de um identificador único para cada usuário, gerado pelo Firebase no cadastro. As informações do usuário já são salvas no banco de dados em um nó identificado por este UUID, tornando-o obrigatório para o acesso às informações e um bom candidato a chave de criptografia.

6.4 Requisitos Funcionais e Não Funcionais - Resultados

Na especificação de requisitos, foram levantados os requisitos funcionais e não funcionais do projeto. Após sua implementação, a quatorze dos requisitos foram cumpridos, um requisito foi parcialmente cumprido, um requisito foi colocado na sprint de backlog e não foi cumprido e um requisito foi retirado do escopo do projeto. Além disso, quatro novos requisitos foram criados e cumpridos ao longo da implementação do projeto.

Cumpridos:

RF01: O usuário deverá conseguir se cadastrar através do aplicativo.
RF02: O professor cadastrado deverá conseguir criar turmas através do aplicativo.
RF03: O aluno cadastrado deverá conseguir ingressar em turmas através do aplicativo.
RF04: O professor poderá adicionar exercícios aos capítulos das turmas.
RF04.2: O exercício poderá ter dificuldade, dada pelo professor.
RF06: O aluno deverá conseguir responder os exercícios propostos através do aplicativo.
RF06.1: O aluno deverá receber do aplicativo, uma resposta de acerto ou erro no exercício resolvido.
RF07: O professor poderá criar programas de progressão em um capítulo, baseado no número de exercícios a serem resolvidos de cada dificuldade.
RNF01: Os aplicativos do professor e do aluno serão separados.
RNF02: A turma poderá ter uma senha de ingresso definida pelo professor.
RNF03: O Aplicativo será desenvolvido para a plataforma Android.
RNF04: O aplicativo se comunicará com o banco de dados não-relacional Firebase.
RNF05: O aplicativo não deverá exceder o tempo de resposta de 1 minuto, quando conectado à internet.
RNF06: Os exercícios podem ser imagens ou textos.

Parcialmente Cumpridos:

RF05: O desempenho do aluno poderá ser visto pelo professor da turma e pelo próprio aluno.

Durante a implementação, esse requisito foi repensado, e apenas o professor terá acesso ao desempenho do aluno.

Não Cumpridos:

RF04.1: O professor poderá adicionar exercícios com horário agendado.

O projeto foi focado para uma ferramenta de acompanhamento do desenvolvimento dos alunos na matéria, e esse requisito é específico para um sistema de avaliação, e foi retirado.

Backlog:

RF01.1: O usuário deverá conseguir mudar sua senha pelo aplicativo.

Outros requisitos foram levantados durante a implementação do projeto, o RF01.1 foi preterido e alocado para a sprint de backlog, por ser indispensável para um produto a ser lançado no mercado, mas não ser essencial para um protótipo e uma prova de conceito.

Novos requisitos realizados:

RNF07: Os dados sensíveis dos usuários serão criptografados.

RF08: O professor poderá copiar capítulos e exercícios de uma turma já existente, em uma turma recém-criada.

RF09: O professor poderá acessar o perfil dos alunos de sua turma, e os alunos poderão acessar o perfil do professor da turma que ingressaram.

RF10: O usuário poderá clicar em uma imagem para visualizá-la melhor.

7 TESTES E AVALIAÇÃO

7.1 Testes Modulares e Globais

Como já citado, o sistema foi desenvolvido utilizando a arquitetura MVC. Cada tela desenvolvida, e o adaptador de criptografia, constituem um módulo MVC independente. Após o desenvolvimento de cada módulo, foram aplicados os testes unitários planejados para o sistema:

- Teste de integridade do aplicativo
- Teste de integridade dos dados
- Teste de funcionalidade

Após a conclusão dos testes unitários de cada módulo, foi realizado um teste de integridade global, em que, com o módulo já acoplado ao sistema, um fluxo completo, simulando um usuário, para garantir a funcionalidade do módulo.

No final de cada sprint, novos testes de integridade global foram realizados, para validade o fim da sprint.

7.2 Performance do Aplicativo e Firebase (Teste de Escalabilidade)

Um aspecto importante desse trabalho é a necessidade de que o *software* desenvolvido seja escalável, e possa ser utilizado por um grande número usuários sem que sua performance seja afetada.

O teste de escalabilidade foi desenvolvido para medir dois parâmetros: Tempo máximo de resposta do aplicativo e quantidade de dados baixada pelo usuário.

Para o teste, um banco de dados foi criado para estressar o sistema. Foi criada uma turma com 100 alunos, 50 capítulos e 500 exercícios por capítulo. Uma turma, com essas configurações ocupou 3,73 Mb no Firebase.

7.2.1 Teste de tempo máximo de resposta:

Durante o teste do aplicativo, detectou-se que o maior tempo de resposta do banco de dados dava-se na tela inicial do aplicativo, quando o usuário estava

logado. Esse tempo de espera deve-se ao fato de ser a primeira conexão do aplicativo com o Firebase.

Com o *smartphone* conectado ao computador, o aplicativo do aluno, e posteriormente o aplicativo do professor, foram abertos dez vezes cada, e o tempo de resposta do Firebase, foi marcado no console do Android Studio e anotado. Em dezenove casos, o tempo de resposta teve uma variação inferior a 5%, e uma média de 2,34 segundos. Em um dos casos, no programa aplicativo do professor, o tempo de resposta foi de 5,88 segundos. Esse último foi descartado, e associado a alguma interferência externa.

7.2.2 Teste de quantidade de dados:

Foram criados dois fluxos de uso para esse teste. Um para o professor, e outro para o aluno. Foi marcado no *smartphone* a quantidade de dados móveis utilizada para realizar cada um dos fluxos. O fluxo do aluno, consistiu em entrar no programa aplicativo, acessar três capítulos diferentes, resolver uma rotina de exercício em cada capítulo, composta por dez exercícios, todos contendo imagens. O gasto de dados móveis foi de 10,2 Mb.

O fluxo do professor consistiu em entrar no programa aplicativo, acessar um capítulo, acessar dez exercícios diferentes, aplicar uma rotina de exercícios e então verificar o desempenho de dez alunos, e da turma inteira. O gasto de dados móveis foi de 6.3 Mb.

A conclusão dos testes de escalabilidade foi de que o *software* desenvolvido é escalável, para a quantidade de dados e usuários projetada.

7.3 Custo Financeiro de Manutenção

O custo financeiro de manutenção é exclusivamente do Firebase, de acordo com fluxo de dados e dados salvos. Um cálculo de custo foi realizado a seguir, considerando uma escola de médio porte. O volume de dados utilizado para cálculo foi estimado baseando-se nos testes realizados com o programa aplicativo.

Número de turmas: 16
Número de turmas base (sem alunos): 8

Volume de dados de uma turma: 2 Mb
Alunos por turma: 50
Alunos total da escola: 400 (cada aluno participa de 2 turmas)
Número de capítulos: 30
Número de exercícios: 500
Proporção de exercícios que utilizam imagens: 20%
Tamanho médio de imagens: 1 Mb
Estimativa de volume de dados salvos por turma: 2 Mb
Estimativa de rotinas realizadas por alunos: 12 por mês
Exercícios por rotina: 6
Estimativa de gasto por exercício com imagem: 300 kb
Estimativa de gasto por exercício sem imagem: 10 kb
Estimativa de gasto mensal do professor: 20 Mb

Volume de fluxo de dados no Realtime Database por mês: 6 * 12 * 10 kb * 0.8 * 16 * 50 + 16 * 20 Mb = 780 Mb
Volume de fluxo de dados no Firebase Storage por mês: 6 * 12 * 300 kb * 0.2 * 16 * 50 = 3,5 Gb
Volume de dados armazenados no Realtime Database: 2 Mb * 24 = 48 Mb
Volume de dados armazenado no Firebase Storage: 500 * 0,2 * 30 * 8 * 1 Mb = 24 Gb

Preço do fluxo de dados no Firebase Realtime Database: U$ 1 / Gb
Preço do fluxo de dados no Firebase Storage: U$ 0,12 / Gb
Preço do volume armazenado no Firebase Realtime Database: U$ 5 / Gb
Preço do volume armazenado no Firebase Storage: U$ 0,026 / Gb

Custo financeiro mensal para manutenção: 0.78 * 1 + 3.5 * 0,12 + 0,048 * 5 + 0,026 * 24 = U$ 2,064

O custo estimado, fica abaixo de R$ 10, em valores convertidos, validando o projeto como solução de baixo custo.

8 CONSIDERAÇÕES FINAIS

8.1 Conclusões do Projeto de Formatura

Este projeto teve como objetivo primário a criação de um protótipo de *software* livre para dispositivos móveis, e que pudesse ser aplicado de maneira simples em salas de aula. Especificamente foi escolhido a aplicação de exercícios a partir de um banco de dados e a possibilidade de o professor acompanhar o desempenho de sua turma.

O resultado obtido foi muito próximo ao planejado. Durante o desenvolvimento alguns requisitos foram despriorizados, e outros surgiram de necessidades detectadas ou avaliações externas do produto.

O sistema desenvolvido cumpre a proposta inicial, demonstrando e expandindo as possibilidades da área de *m-learning*. Apesar de não estar pronto para pleno uso, esse projeto cumpriu o papel de desenvolver uma ferramenta viável para ampla utilização.

Ampliando algumas funcionalidades, e melhorando alguns aspectos de escalabilidade e principalmente experiência do usuário, o *software* desenvolvido pode no futuro ter um papel importante na crescente integração entre a sala de aula e a tecnologia de aplicações móveis.

8.2 Contribuições

Na elaboração deste trabalho, foram utilizados códigos de software livre, desenvolvidos por terceiros, para Android a saber:

- O *Glide* foi utilizado para a instanciação de imagens através de um *Uniform Resource Locator* (URL):
  https://github.com/bumptech/glide

- O *Material Spinner* foi utilizado para a criação dos menus colapsáveis do projeto:
  https://github.com/jaredrummler/MaterialSpinner

- O *Android Image Popup* foi utilizado para o usuário conseguir abrir as imagens dos exercícios clicando sobre elas: https://github.com/chathuralakmal/AndroidImagePopup

8.3 Perspectivas de Continuidade

Para a continuidade do projeto, há três aspectos que podem ser continuados e expandidos no futuro: Finalização do produto, expansão de plataformas e incremento de funcionalidades.

8.3.1 Finalização do produto:

O projeto, como foi desenvolvido, trata-se de um protótipo, e por esse motivo, há algumas deficiências quanto ao seu *design* e experiência do usuário.

Outro ponto em continuidade ao projeto é a implementação de rotinas básicas, necessárias para o uso do *software*, e que justamente por serem rotinas de finalização de produto, foram alocadas para o *backlog* do projeto. São os métodos de mudança de senha, edição de perfil e edição de turma.

8.3.2 Expansão de plataformas:

Um aspecto discutido durante o desenvolvimento do projeto foi a expansão do sistema para outras plataformas além de Android. Duas plataformas foram planejadas para expansão: web e iOS.

Expansão para iOS é importante pois é o sistema operacional mais utilizado em aparelhos móveis após Android. Hoje, 15% dos dispositivos móveis utilizam o iOS (IDC, 2018) e como os alunos devem executar o programa aplicativo, a compatibilidade com aparelhos móveis deve ser próxima a 100%. Por tratar-se de uma prova de conceito, esse trabalho focou em implementar um protótipo funcional, evitando a duplicação para do aplicativo para iOS, ou a implementação através de uma plataforma para os dois sistemas, porém mais custosa e menos documentada.

A expansão para a web surgiu como ponto importante de melhoria, detectado ao longo do desenvolvimento e de testes com usuários. A interface de criação de

turmas, especialmente criação de exercícios, foi apontada como deficiente, por ser muito mais trabalhoso selecionar imagens e copiar textos no *smartphone*. A interface existente deve ser mantida, para preservar as vantagens que dispositivos móveis fornecem, mas a criação de uma plataforma web, exclusivamente para montar as turmas, é um ponto importante de melhora.

8.3.3 Incremento de funcionalidades:

O último ponto de melhoria para o projeto é a implementação de novas funcionalidades. O objetivo deste trabalho foi a criação de uma ferramenta de uso simples mas com potencial de impactar no ensino, e resolver um problema específico e definido. A implementação de novas funcionalidades deve ter como foco a melhora na resolução deste problema, e não na solução de novos.

Funcionalidades possíveis levantadas ao longo do desenvolvimento foram: (i) possibilitar mais de um professor por turma; (ii) possibilitar a visualização do desempenho por rotinas realizadas, podendo o professor acompanhar a evolução do aluno e a possibilidade de o professor definir uma data máxima para resolução do exercício; e (iii) a tela de análise de desempenho mostrar os exercícios não resolvidos pelo aluno.

REFERÊNCIAS

AMAZON. <https://aws.amazon.com/pt/> Acesso em: 10 abr. 2018

CHAVES, E. Tecnologia na educação. 2004. Disponível em: <https://smeduquedecaxias.rj.gov.br/nead/Biblioteca/Forma%C3%A7%C3%A3o%20Continuada/Tecnologia/chaves-tecnologia.pdf> Acesso em: 20 set. 2018.

DRIGAS, A.; ANGELIDAKIS, P. Mobile Applications within Education: An Overview of Application Paradigms in Specific Categories, iJIM – Vol. 11, No. 4, 2017, pp. 17-29

FIRB. Trabalhos acadêmicos: Normas da ABNT. Disponível em: <http://www.firb.br/abntmonograf.htm>. Acesso em 04 abr. 2018.

GARCIA, L. F. F. Engenharia de Software I. 1ª. ed. Canoas: ULBRA, 2013.

GARCIA, P.; MARTINI, J. S. Ensino on-line e laboratório - ensinando, avaliando e certificando. São Paulo: LivroPronto Editora e Gráfica, 2007.

GOOGLE. <https://support.google.com/edu/classroom/?hl=pt-BR#topic=6020277> Acesso em: 25 out. 2018

MICROSOFT. <https://docs.microsoft.com/pt-br/xamarin/> Acesso em: 10 abr. 2018.

NAIL, B, AMMAR, W. A. Mobile Learning Education has Become More Accessible. Am J Compt Sci Inform Technol 5:2, 2017.

TEACHERKIT <http://teacherkit.net/> Acesso em: 10 abr. 2018

TRABALHOS ABNT. Regras e Normas da ABNT 2012 para formatação de trabalhos acadêmicos. Disponível em:<http://www.trabalhosabnt.com/regras-normas-abnt-formatacao>. Acesso em: 04 abr. 2018.

UNITY. <https://unity3d.com/pt>. Acesso em: 10 abr. 2018.

WALKER, R. “I don't think I would be where I am right now". Pupil perspectives on using mobile devices for learning', Research in Learning Technology, vol. 21, 2013, pp. 1-12.

WIKIMEDIA. Normas ABNT para apresentação de trabalhos científicos. Disponível em:<http://pt.wikipedia.org/wiki/Normas_ABNT_para_apresentação_de_trabalhos_científicos>. Acesso em: 04 abr. 2018.