UNIVERSIDADE COMUNITÁRIA DA REGIÃO DE CHAPECÓ
ÁREA DE CIÊNCIAS EXATAS E AMBIENTAIS
CIÊNCIA DA COMPUTAÇÃO
(BACHARELADO)

APLICATIVO MÓVEL PARA O SISTEMA ACADÊMICO MINHA UNO

ANDREI JIÁCOMO ZUSE

CHAPECÓ, JUNHO DE 2013

UNIVERSIDADE COMUNITÁRIA DA REGIÃO DE CHAPECÓ
ÁREA DE CIÊNCIAS EXATAS E AMBIENTAIS
CURSO DE CIÊNCIA DA COMPUTAÇÃO
(BACHARELADO)

APLICATIVO MÓVEL PARA O SISTEMA ACADÊMICO MINHA UNO

Relatório do Trabalho de Conclusão de Curso submetido à Universidade Comunitária da Região de Chapecó para obtenção do título de bacharel em Ciência da Computação.

ANDREI JIÁCOMO ZUSE

Orientador(a): Prof. Marcelo Cezar Pinto, Me.

CHAPECÓ, JUNHO DE 2013

APLICATIVO MÓVEL PARA O SISTEMA ACADÊMICO MINHA UNO

ANDREI JIÁCOMO ZUSE

Este trabalho de conclusão de curso foi julgado adequado para obtenção de título de graduação em 25 de Junho de 2013, e foi aprovado pelo curso de bacharelado em ciência da computação da Universidade Comunitária da Região de Chapecó

Prof. Marcelo Cezar Pinto, Me.
Orientador

BANCA EXAMINADORA:

Prof. Marcos Antonio Moretto, Esp.
Unochapecó

Prof. Jean Carlos Hennrichs, Esp.
Unochapecó

Prof. Sandro Silva de Oliveira, Me.
Supervisor de TCC

Profa. Viviane Duarte Bonfim, Ma.
Coordenadora de Curso

Chapecó, 25 de Junho de 2013

RESUMO

Este trabalho apresenta um aplicativo para dispositivos móveis para o sistema acadêmico Minha Uno, permitindo que os acadêmicos da universidade acessem por meio de um aplicativo as informações da sua graduação. Utilizando-se de técnicas de extração de dados a partir do código HTML da página do sistema acadêmico Minha Uno e implementando um servidor REST, posteriormente consumido pela aplicação. Desta forma o trabalho divide-se em duas linhas práticas, abordando o servidor de extração de informações do sistema acadêmico, e outra onde é exibida a implementação da aplicação, consumindo as informações do servidor implementado e mostrando ao usuário em seu dispositivo com o sistema operacional Google Android ou iOS. Os módulos implementados foram escolhidos por meio de questionário online aplicado aos acadêmicos, assim como as plataformas a serem contempladas com a aplicação e outras informações sobre o perfil dos acadêmicos que utilizam o Sistema Acadêmico Minha Uno. Ao final do trabalho é possível concluir que o servidor implementado pode fornecer informações não apenas para o aplicativo móvel, mas também para outros aplicativos que venham a ser desenvolvidos, e que a implementação de um aplicativo para dispositivos móveis é de desejo da comunidade acadêmica e é viável a sua implementação.

Palavras-chave: Aplicativo Móvel; Extração de Informações; Sistema Acadêmico; Unochapecó

ABSTRACT

This paper presents a mobile application for the academic system Minha Uno, allowing university academics to access through an application the information of their graduation. Using techniques of data extraction from the HTML page of the academic system Minha Uno and implementing a REST server subsequently consumed by the application. Thus the work is divided into two practices lines, one addressing the de data extraction server from academic system, and another which opens the implementation of the application, consuming the information server implemented and showing the user on your device with the operating system Google Android or iOS. The modules implemented were chosen through online questionnaire applied to academics, as well as the platforms to be included with the application and other information about the profile of the students who use the Academic System Minha Uno. At the end of the study it can be concluded that the implemented server can provide information not only for the mobile application, but also for other applications that may be developed, and that the implementation of an application for mobile devices is the desire of the academic community and is feasible implementation.

Keywords: Mobility; Server; Extraction; Communication; Unochapecó

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

A adoção de dispositivos móveis com sistema operacional Android e iOS chegam a níveis nunca antes vistos na história da tecnologia. Comparada com outras tecnologias recentes, a adoção de dispositivos do tipo smart foi dez vezes mais rápida que a revolução dos Computadores Pessoais nos anos 80, duas vezes mais rápida que a explosão da Internet nos anos 90 e três vezes mais rápida que a adoção de Redes Sociais. Estima-se que, no mundo, existam 640 milhões de dispositivos com iOS e Android em uso durante o mês de Julho de 2012 (FLURRY, 2012).

No período de Julho de 2011 a Julho de 2012, o Brasil apresentou o 3º maior crescimento no número de dispositivos móveis, com um aumento de 220%. Estima-se que o Brasil possua 13 milhões de dispositivos com iOS e Android ativos, ocupando o décimo lugar no ranking mundial de dispositivos móveis ativos (FLURRY, 2012).

Em dados da Huawei (fabricante dos equipamentos utilizados pelas operadoras de telefonia móvel) em seu balanço anual, no Brasil houve um aumento de 99% no número de usuários da tecnologia 3G. Até o mês de Abril do ano de 2012, em relação ao final de 2011, ocorreu um aumento de 31% no número de usuários e levando-se em consideração o intervalo do primeiro trimestre de 2011 até o primeiro trimestre de 2012, houve um aumento de 112,6% no número de usuários (HUAWEI, 2012).

Considerando válida a suposição de que a expansão e uso de dispositivos móveis e tecnologia 3G é similar ao do Brasil na região de Chapecó, pode-se verificar pelos números do balanço social 2011 da Fundação Universitária do Desenvolvimento do Oeste (FUNDESTE), instituição mantenedora da Universidade Comunitária da Região de Chapecó (Unochapecó), que o número de pessoas vinculadas a universidade com acesso a estas tecnologias é significativo. Segundo o balanço social 2011, a Unochapecó conta com 947 funcionários sendo destes 540 docentes e 407 técnico-administrativos. Segundo este mesmo documento, a instituição conta com 8031 acadêmicos de graduação e 910 acadêmicos de pós-graduação, totalizando 8941 acadêmicos ao final do ano de 2011 (FUNDESTE, 2011). Atualmente todas estas pessoas, acadêmicos ou funcionários da Universidade utilizam uma página web para acessar as informações do(s) seu(s) perfil(s), chamada “Minha Uno”. Deste universo de quase 9 mil pessoas, pode-se estimar com base nos dados anteriormente vistos que mais de 600 pessoas envolvidas com a universidade possuam smartphones ou tablets.

1.1 Tema

Devido a crescente utilização de dispositivos móveis (tablets e smartphones) no meio acadêmico, e a facilidade de acesso a internet sem fio e móvel, pretende-se com este trabalho melhorar o acesso ao sistema acadêmico da Universidade Comunitária da Região de Chapecó (Unochapecó) nos mesmos, por meio do desenvolvimento de um aplicativo que notifique o usuário sobre novas informações no sistema e também permita a consulta aos dados do sistema conforme suas permissões, além de poder efetuar cadastros.

1.2 Delimitação do Problema

Tendo como ponto de partida o funcionamento atual do Sistema Acadêmico (Minha Uno), existe uma forma de melhorar a forma de acesso em dispositivos móveis por meio de um aplicativo, agregando recursos que auxiliem os usuários do mesmo?

1.3 Questões de Pesquisa

Quais funcionalidades do sistema acadêmico os usuários desejam que esteja presente em seus dispositivos móveis?

Qual tecnologia de desenvolvimento multiplataforma para dispositivos móveis se adapta melhor as necessidades de desenvolvimento da aplicação?

Qual será a forma de comunicação com o sistema acadêmico atual, para a coleta das informações necessárias para o funcionamento nos dispositivos móveis?

Quais as plataformas que serão contempladas com esta aplicação?

Como permitir que o usuário acesse as informações sem estar conectado a internet?

1.4 Objetivos

1.4.1 Objetivo Geral

Melhorar a forma de acesso ao Sistema Acadêmico em dispositivos móveis por meio de um aplicativo para consulta das informações pertencentes ao perfil do usuário, além de notificações sobre alterações no sistema que sejam de interesse do utilizador da aplicação.

1.4.2 Objetivos Específicos

Definir por meio de um questionário aplicado aos usuários do Sistema Acadêmico quais funcionalidades são importantes para cada perfil de usuário existente no sistema, e posteriormente definir quais destas funcionalidade estarão disponíveis no aplicativo.

Pesquisar qual tecnologia de desenvolvimento multiplataforma fornece os recursos necessários para o desenvolvimento da aplicação.

Definir juntamente com a equipe de Tecnologia da Informação da instituição a forma de comunicação com o Sistema Acadêmico, assim como quais informações serão disponibilizadas pela instituição.

Por meio do questionário efetuado com os usuários do Sistema Acadêmico, levantar as plataformas móveis mais utilizadas para desenvolver a aplicação.

Utilizando uma base de dados no dispositivo móvel, fazer com que as informações do acadêmico possam ser consultadas sem conexão com a internet.

1.5 Justificativa

A utilização de dispositivos móveis vem batendo records de crescimento a cada ano, superando outras inovações tecnológicas consideradas fundamentais nos dias de hoje, como os Computadores Pessoais, a Internet e as Redes Sociais, estimando-se que existam 640 milhões de dispositivos móveis com sistema operacional Android ou iOS sendo utilizados no mundo durante o mês de Julho de 2012. O Brasil apresentou o terceiro maior crescimento na adoção deste tipo de dispositivos entre 2011 e 2012, tendo um crescimento de 220% no número de usuários, ficando atrás da China com aumento de 401% e do Chile com aumento de 279% neste mesmo período. Além disto, o Brasil encontra-se em décimo lugar no ranking de dispositivos móveis com Sistema Operacional Android ou iOS, com 13 milhões de dispositivos ativos, atrás dos Estados Unidos com 165 milhões de dispositivos, China com 128 milhões, Reino Unido com 31 milhões, Coréia do Sul com 28 milhões de dispositivos, Japão com 22 milhões de dispositivos, Alemanha com 19 milhões de dispositivos, França com 17 milhões de dispositivos, Canadá com 16 milhões de dispositivos e Espanha com 13 milhões de dispositivos (FLURRY, 2012).

Também percebe-se aumento de 26,2% na utilização de internet banda larga 3G no mundo no período de 2010 até 2011. No Brasil houve um aumento mais significativo, chegando a 99% no mesmo período, passando de 20,6 milhões de usuários em 2010 para 41,1 milhões de usuários em 2011. Tendo como período para avaliação desde o primeiro trimestre de 2011 até o primeiro trimestre de 2012, percebe-se um aumento de 112,6% no número de usuários de conexão banda larga 3G no país (HUAWEI, 2012).

Apesar do grande aumento no número de usuários destas tecnologias, o país ainda não se encontra saturado de dispositivos móveis, como ocorre, por exemplo, em Singapura, onde 92% das pessoas entre 15 e 64 anos possuem um dispositivo móvel com Android ou iOS, tendo assim um crescimento anual discreto no número de usuários devido a esta saturação. Nos Estados Unidos o número de pessoas entre 15 e 64 anos que possuem este tipo de dispositivo totalizam 310 milhões de pessoas, 78% da população esta faixa etária (FLURRY, 2012).

Em uma pesquisa semelhante feita por acadêmicos da Universidade de Minho (localizada em Braga, Portugal) durante o ano de 2008, constatou-se que entre as 1225 pessoas de diferentes centros de ensino superior de Portugal que participaram da pesquisa, apenas 1% das pessoas pesquisadas não possuem dispositivos móveis (*Personal Digital Assitant* (PDA),

smartphones). Levando-se em consideração o crescimento das plataformas móveis nos anos posteriores a este período, com o lançamento da primeira geração do iPhone no ano de 2008 em Portugal, e posteriormente a popularização da plataforma Android nos anos subsequentes, mostra que o uso de Celulares, smartphones, PDA's (e atualmente tablets) é popular nos meios acadêmicos deste país (JUNIOR; COUTINHO, 2008). Devido a Portugal ser um país desenvolvido e o Brasil ser um país em desenvolvimento, esta pesquisa pode não refletir a quantidade real da utilização nas instituições de ensino superior no Brasil ou da Unochapecó, mas demonstra que a utilização de dispositivos móveis em meios acadêmicos é comum, ficando até mesmo acima da média nacional.

Por meio de uma pesquisa semelhante a realizada em Portugal, sendo esta aplicada na Unochapecó durante o desenvolvimento deste trabalho, será possível definir o percentual de usuários dentre os pesquisados que utilizam dispositivos móveis para acesso ao sistema acadêmico, e também definir quais as funcionalidades do Sistema Acadêmico Minha Uno são mais importantes para os usuários de cada perfil, podendo assim servir como base para o desenvolvimento da aplicação, atendendo diretamente as necessidades dos usuários do mesmo.

Tendo como ponto de partida a escolha das funcionalidades do sistema, é necessário definir as ferramentas utilizadas na implementação do aplicativo. Para isto, deve-se avaliar as três categorias de ferramentas de desenvolvimento para mobilidade multiplataforma existentes (Compilação Cruzada, Máquina Virtual e Webkit) para definir qual melhor atente das necessidades no desenvolvimento da aplicação. Cada classe de ferramentas possui um propósito, possuindo pontos positivos e negativos a serem levados em consideração durante o desenvolvimento da aplicação(HARTMANN; STEAD; DEGANI, 2011).

Para efetuar-se a integração entre aplicativos para dispositivos móveis e páginas web, são disponibilizados web services, que utilizando de protocolos como o *Simple Object Access Protocol* (SOAP), *Web Services Description Language* (WSDL), *Universal Description Discovery and Integration* (UDDI) (ou outras tecnologias mais recentes como o *JavaScript Object Notation* (JSON)), disponibilizam informações via Linguagem Extensível de Marcação (XML) para outras aplicações (SHKLAR; ROSEN, 2003). Tem-se como exemplo disto o Facebook, que disponibiliza uma *Application programming interface* (API) de desenvolvimento, que extrai informações dos webservices, e também permite inserção de novas informações utilizando-se destas mesmas ferramentas (FACEBOOK, 2012).

1.6 Procedimentos Metodológicos

Os métodos utilizados para a elaboração deste projeto serão feitos através de pesquisa bibliográfica, artigos retirados da internet, coleta de dados feita através de questionário e desenvolvimento de um protótipo de acesso ao sistema para dispositivos móveis.

Quanto à coleta de dados, será feito um questionário com alunos, professores e funcionários da Unochapecó, para levantar informações sobre os usuários do Sistema Acadêmico Minha Uno. Após o término do questionário e interpretação dos dados coletados, serão determinadas quais as funcionalidades do sistema acadêmico são mais relevantes para seus usuários, percentual de usuários que possuem smartphones, tablets ou ambos, os sistemas operacionais móveis mais utilizados pelos mesmos e a principal forma de acesso a internet utilizada em seus dispositivos móveis. A partir destes dados será possível determinar quais as prioridades no desenvolvimento da aplicação e a divisão das etapas de desenvolvimento da mesma, tendo em vista nas etapas iniciais as funcionalidades mais importantes para cada perfil de usuário.

A pesquisa bibliográfica no aspecto de desenvolvimento para dispositivos móveis e integração com webservice, conta com um número considerável de obras e autores. Portanto, este trabalho acadêmico, neste ponto, terá suas necessidades supridas.

1.7 Estrutura do Trabalho

No primeiro capítulo deste trabalho é apresentada uma introdução ao trabalho, descrevendo o tema escolhido e os objetivos a serem alcançados. Também são descritos os procedimentos metodológicos utilizados no desenvolvimento do mesmo.

No segundo capítulo deste trabalho são apresentados os conceitos das tecnologias aplicadas neste trabalho, apresentando os sistemas operacionais para dispositivos móveis e as linguagens de programação e bibliotecas utilizadas no desenvolvimento do trabalho.

No terceiro capítulo deste trabalho são apresentadas as diversas categorias de ferramentas de desenvolvimento para dispositivos móveis, explicando seu funcionamento e processo de geração da aplicação.

No quarto capítulo deste trabalho são apresentos os meios de comunicação aplicados no trabalho, explicando o funcionamento das redes sem fio e da telefonia móvel, explicando as diversas tecnologias de telefonia móvel existentes.

No quinto capítulo deste trabalho é apresentada uma visão geral do sistema acadêmico Minha Uno, com informações sobre os seus perfis de acesso.

No sexto capítulo deste trabalho são apresentados os dados da pesquisa feita com os usuários do sistema acadêmico sobre a utilização de dispositivos móveis e a importância dos módulos do sistema acadêmico.

No sétimo capítulo é explicado o desenvolvimento do servidor utilizado para a extração das informações do sistema acadêmico Minha Uno, e também como as informações são disponiblizadas para a aplicação móvel.

No oitavo capítulo é explicado o desenvolvimento da aplicação para dispositivos móveis, demonstrando a forma de desenvolvimento utilizada e o resultado apresentado na aplicação.

No nono capítulo é feita a conclusão do trabalho, e são sugeridos temas para a continuidade do mesmo posteriormente.

2 TECNOLOGIAS

2.1 Sistema Operacional para Dispositivos Móveis

Sistemas operacionais para dispositivos móveis seguem o mesmo conceito aplicado a outras plataformas, que segundo Tanembaum, é de difícil definição por meio de um único conceito, pois o mesmo é responsável pelo Gerenciamento do Hardware hospedeiro do mesmo, como também fornecer um conjunto de recursos abstratos claros em vez de recursos confusos de hardware aos aplicativos (TANENBAUM, 2010). Em outras palavras, pode-se dizer que o sistema operacional é formado de um conjunto de softwares e bibliotecas que tornam mais fácil a interação entre a máquina e os usuários.

2.1.1 Symbian

Desenvolvido pela parceria entre Nokia, Ericsson, Motorola e PSION, foi usado amplamente pela Nokia em praticamente todos os seus dispositivos no passado. O sistema operacional tinha como foco a integridade e segurança de dados, evitar desperdício de tempo do usuário e trabalhar com recursos escassos. Era reconhecido pelo gerenciamento extremamente eficiente de recursos como bateria, processador e memória(CARVALHO et al., 2010).

Devido a concorrência de outros sistemas operacionais para dispositivos móveis, e por ter uma diminuição significativa nas suas vendas, o Symbian foi descontinuado pela Nokia, a principal investidora e utilizadora destes sistema operacional, anunciando em janeiro de 2013 a não utilização do sistema em seus aparelhos, substituindo o mesmo pelo Windows Phone nos smartphones da companhia (NOKIA, 2013).

2.1.2 iOS

Desenvolvido pela Apple, tendo como base o sistema operacional Mac OS X, o iOS inicialmente foi desenvolvido para iPhone, depois expandido para outros produtos como iPad e iPod. É o único sistema operacional analisado desenvolvido para dispositivos específicos, sendo totalmente fechado aos mesmos. Apresenta um ótimo desempenho devido a integração entre Hardware e Software, além da interface gráfica intuitiva, porém apresenta dificuldades para interagir com outros dispositivos externos (como por exemplo outros celulares utilizando a tecnologia Bluetooth) (COSTA; FILHO; DUARTE, 2012).

Atualmente o iOS encontra-se na sua sexta versão, e mantém as suas principais qualidades desde a primeira versão, sendo destacadas a interface gráfica intuitiva, recursos, facilidade da utilização e estabilidade do sistema. Além disso, é possível atualizar facilmente um dispositivo iOS quando uma nova versão é lançada, sendo todo processo feito pelo próprio dispositivo caso o usuário assim deseje (APPLE, 2013).

Devido ao fato de a Apple produzir tanto o hardware como o software, o sistema operacional é totalmente compatível com o hardware a ser instalado, disponibilizando os recursos conforme o hardware de cada dispositivo. Além disso, a Apple preocupa-se com a segurança e estabilidade do sistema, implementando meios de proteção física contra vírus e malwares, e via aplicação para proteger os dados do usuários (APPLE, 2013).

2.1.3 Google Android

O Google Android surgiu da necessidades de várias empresas fabricantes de Hardware, Software e Operadoras de Telefonia Móvel de um sistema operacional que apresentasse vantagens para os desenvolvedores de Software e uma experiência de usuário inovadora, com muitos recursos sem abrir mão da beleza e da facilidade de uso. Desenvolvido pela Open Handset Alliance (OHA, 2012), o Google Android é um sistema operacional desenvolvido sobre o kernel Linux com aplicações nativas e a possibilidade desenvolvimento de aplicações por qualquer pessoa que possua as habilidades e vontade para isso (LECHETA, 2009).

Atualmente o Google Android encontra-se na sua versão 4 revisão 2, e apresenta como novidades desta versão a possibilidade de os usuários do Sistema Operacional que utilizam o mesmo em Tablets possam criar perfis diferentes, com aplicativos e configurações diferentes.

Além disso o Google trouxe ao Android recursos de transmissão de imagens sem fio e melhorias no desempenho do sistmea operacional em geral. Para completar foram implementados novos recursos para a câmera e também um teclado melhorado (GOOGLE, 2013).

2.1.4 Windows Phone

Lançado pela Microsoft em 2010, como sucessor do Windows Mobile, o Windows Phone foi desenvolvido para atender tanto usuários comuns como corporativos. O Windows Phone foi desenvolvido para rodar em diversos hardwares, podendo ser licenciado pelas fabricantes que desejarem usar o mesmo em seus dispositivos. Possui integração total com o ambiente .NET também da Microsoft, além das tecnologias de desenvolvimento Silverlight e XNA. Apresenta uma desvantagem em relação aos outros sistemas operacionais que é não possuir multithread, permitindo que apenas uma aplicação seja executada por vez no dispositivo (COSTA; FILHO; DUARTE, 2012).

2.2 Web service

Web service é um sistema de software que permite a comunicação entre diversos softwares por meio de mensagens SOAP, normalmente utilizando o Protocolo de transferência de Hipertexto (HTTP) para transmitir arquivos de dados entre as aplicações (W3C, 2012).

2.2.1 REST

A arquitetura de Transferência de Estado Representativo (do inglês *REpresentational State Transfer*, também identificado como REST) é uma técnica da engenharia de software que define uma interface uniforme de conexão. O REST é um estilo híbrido, baseado em muitos protocolos baseados em rede (como Protocolo Cliente-Servidor, Protocolo de Sistemas em Camadas, Protocolo de Sessão Remota, etc.) (FIELDING, 2000).

A arquitetura REST funciona com o conceito Cliente-Servidor, onde as informações são disponibilizadas em um servidor, que responde de forma padronizada as requisições de informações, retornando com um mesmo layout os dados retornados. Isto permite que várias aplicações venham a consumir informações do mesmo servidor, porém mostrem estas informações de diferentes formas, já que o servidor REST não retorna componentes gráficos, apenas

texto em algum padrão definido (exemplo XML, JSON, Texto delimitado por caractere) (FIELDING, 2000).

Um servidor REST é um servidor sem estado, ou seja, as informações retornadas pelo servidor não são armazenadas no mesmo. Desta forma a visibilidade, confiabilidade e escalabilidade do servidor são melhoradas. A melhoria da visibilidade ocorre pois o servidor não precisa olhar para outros estados em busca de dados, olhando apenas a requisição feita para o mesmo e retornando as informações referentes a ela. Já a confiabilidade melhora pois por ser sem estado o servidor consegue recuperar-se de falhas parciais de forma automática, onde ao se repetir a requisição o erro ocorrido anteriormente não influencia uma nova consulta. Já a escalabilidade é melhorada pois os recursos alocados no servidor para cada consulta são desalocados rápidamente após o termino da mesma, não mantendo nenhuma informação daquela consulta em memória temporária ou permanente. A desvantagem de ser uma arquitetura sem estado é que requisições repetitivas de informações sempre serão buscadas na fonte dos dados, podendo gerar sobrecarga na rede ou em servidores para tratamento destas requisições.(FIELDING, 2000).

Além disso um servidor REST tem como uma das suas principais características a padronização na disponibilização das informações, onde aplicando-se os conceitos da engenharia de software de generalização é possível obter de diferentes servidores REST que possuem como base o mesmo tipo de retorno informações diferentes sem alterar-se a forma de obtenção das informações. Desta forma uma vez desenvolvido o servidor, o consumo das informações ocorre de forma rápida e simples pela aplicação, por seguir um padrão já conhecido de retorno. Este padrão de retorno, conforme comentado anteriormente, não ocorre de forma estática, e é desenvolvido pela implementação do servidor (FIELDING, 2000).

2.3 JavaScript

Apesar do nome possuir a palavra *Java*, o JavaScript não possui ligação nenhuma com esta linguagem de desenvolvimento, sendo o nome escolhido desta forma apenas para causar a confusão e a curiosidade dos desenvolvedores. O JavaScript é uma linguagem de programação dinâmica e orientada a objetos para propósitos gerais, sendo seu principal uso no desenvovimento de scipts para a internet. Atualmente qualquer computador possui um interpretador JavaScript instalado, seja ele integrado ao navegador web ou uma aplicação externa (CROCKFORD, 2001).

O JavaScript foi desenvolvido inicialmente pela Netscape com o nome de LiveScript, porém o nome nõa era confuso o suficiente, sendo alterado para JavaScript posteriormente. A sintaxe das aplicações desenvolvidas em JavaScript lembram de uma aplicação desenvolvida em Java, e consequentemente de uma aplicação desenvolvida em C, porém o JavaScript não possui estruturas de tipos, e apesar de suportar orientação a objetos também não possui classes, o que também mostra que não possuí herança por classes, porém possui herança orientada a protótipos (CROCKFORD, 2001).

O fato de o JavaScript ser uma linguagem com uma curva de aprendizado suave faz com que ele seja utilizado em meio acadêmico para aprendizagem, o que faz com que a linguagem seja conhecida pela sua simplicidade, não pelo seu potencial. Além disso a especificação da linguagem feita pela *ECMA* (Europea Computer Manufactures Association) de uma qualidade muito baixa e de dificil leitura auxiliou no mal entendimento do funcionamento e das especificações da linguagem. Unindo este fato a livros escritos por pessoas que não conhecem a fundo a linguagem e assim demonstram exemplos que não seguem as melhores práticas de desenvolvimento, gera-se um mito de que o JavaScript é uma linguagem desorganizada e que não existe um padrão.(CROCKFORD, 2001).

2.3.1 JSON

Baseando-se em um subconjunto da linguagem JavaScript, o JSON (Notação de Objetos JavaScript, do inglês JavaScript Object Notation) é um formato leve para troca de dados. Sendo completamente independente da linguagem JavaScript, apenas apresentando similaridades com a mesma, é um formato familiar a maioria das linguagens de programação modernas, sendo assim considerado o formato ideal para troca de dados. O JSON representa basicamente dois tipos de estruturas de dados, objetos e vetores.(CROCKFORD, 2013).

Os objetos são representados entre chaves (iniciando em { e terminando em }) e representando os atributos no formato chave-valor, onde a chave obrigatóriamente apresenta-se entre aspas (") e é separada do valor por dois-pontos (:). Os valores podem ser do tipo vetor, objeto, booleano, nulo, texto e número. Todos os atributos de um objeto são separados por vírgula (,), sendo que o último atributo da classe não leva vírgula antes do fechamento das chaves. Um exemplo de objeto JSON é pode ser visto na Figura 1 (CROCKFORD, 2013).

Na Figura 1 também é exibido um vetor, representado pelo atributo *sub1* do objeto

FIGURA 1 - Exemplo de Objeto JSON

```
{
    "FirstObject": {
        "attr1": "one value",
        "attr2": "two value",
        "sub": {
            "sub1": [
                {
                    "sub1_attr": "sub1_attr_value"
                },
                {
                    "sub1_attr": "sub2_attr_value"
                }
            ]
        }
    }
}
```

Fonte: (MAYANI, 2011).

sub. Como pode-se ver, um vetor tem como característica estar entre colchetes (iniciando em [e terminando em]), e seus objetos ou valores separados por vírgula (,), exceto o último valor antes dos colchetes serem fechados (CROCKFORD, 2013).

2.4 Java

Criado por James Gosling a partir de um projeto de pesquisa da Sun Microsystem em 1991, chamado inicialmente de Oak e posteriormente renomeado para Java pois o nome Oak já pertencia a uma linguagem de programação.(DEITEL; DEITEL, 2010).

Baseada na linguagem C++ e orientada a objetos, foi criada para ser utilizada em dispositivos eletrônicos, porém com a expansão da *Web* a Sun viu o potencial de utilizar o Java para adicionar conteúdo dinâmico as páginas *Web*. A linguagem foi anunciada oficialmente em 1995 em uma feira do setor (DEITEL; DEITEL, 2010).

2.5 SQLite

O SQLite é uma biblioteca de código fonte livre compilada juntamente com a aplicação que implementa um controlador de base de dados SQL para uso geral. Por funcionar embutida

na aplicação, não necessita de um servidor ou configurações para funcionar, bastando definir o nome do arquivo da base e a biblioteca se encarrega dos controles necessários (SQLITE, 2013).

Apesar de todas as funcionalidades fornecidas pelo SQLite, como a possibilidade de execução de consutas SQL, gerenciamento de transações e múltiplas tabelas, indices e gatilhos, o SQLite é econômico em termos de memória utilizada, sendo sua utilização recomendada em dispositivos com pouco espaço de armazenamento ou memória ram, como celulares, players mp3 entre outros. Mesmo neste tipo de ambientes, o SQLite apresenta um bom desempenho (SQLITE, 2013).

2.6 jsoup

O jsoup é uma biblioteca para a linguagem java que fornece uma API para extração e manipulação de dados a partir de código HTML utilizando o melhor do DOM, CSS e métodos jquery para extrair as informações (HEDLEY, 2013).

O jsoup permite a extração das informações acessando o código HTML a partir de uma URL, um arquivo ou de um texto, sendo que para a busca e extração das informações podem ser utilizados DOM ou seletores CSS. Além disso o jsoup permite manipular os elementos, atributos e textos dos códitos HTML (HEDLEY, 2013).

O jsoup foi desenvolvido para permitir a extração dos mais diferentes formatos de código HTML disponíveis, validando o HTML de entrada e criando uma árvore informações que posteriormente são extraídas conforme a expressão indicada na extração(HEDLEY, 2013).

3 FERRAMENTAS DE DESENVOLVIMENTO MULTIPLATAFORMA PARA DISPOSITIVOS MÓVEIS

Um dos grandes desafios para os desenvolvedores de software é encontrar qual ferramenta atende as necessidades da aplicação a ser desenvolvida. Isso fica ainda mais perceptível quando a aplicação deve ser executada em dispositivos móveis, devido a diversidade de sistemas operacionais a serem atendidos. Hoje existem 4 sistemas operacionais para dispositivos móveis que detém grande parte do mercado, sendo eles Android, iOS, Windows Phone e Symbian, sendo o Android e o iOS detentores da maior fatia do mercado. Ao se deparar com este cenário, desenvolvedores se perguntam qual tipo de ferramenta permite que eles desenvolvam apenas uma vez a aplicação, e que a mesma rode de forma satisfatória em todos os dispositivos. Abaixo será visto as 3 principais técnicas de desenvolvimento multiplataforma para dispositivos móveis(HARTMANN; STEAD; DEGANI, 2011).

3.1 Compilação Cruzada

Ferramentas do tipo Compilação Cruzada separam os ambientes de compilação conforme a plataforma de destino, gerando código fonte nativo para a plataforma de destino neste momento do processo de geração de um aplicativo. Com isso são geradas aplicações que são nativas para as plataformas alvo. Este tipo de ferramenta possui a desvantagem de consumir mais tempo durante a compilação, devido a conversão de código, e por apresentar certa demora para aderir a novas plataformas (HARTMANN; STEAD; DEGANI, 2011).

3.1.1 Titanium

Desenvolvido pela Appcelerator, o Titanium foi lançado em dezembro de 2008. Baseado na técnica de compilação cruzada, o desenvolvedor utiliza-se de uma API JavaScript para o

desenvolvimento da aplicação, e no momento da compilação são gerados códigos nativos para a diferentes plataformas. A compilação é composta por 3 etapas, sendo elas: Pré compilação (Otimização do código JavaScript), Compilação de Front-End (Geração de código nativo da plataforma alvo) e Compilação e Empacotamento para a plataforma (Geração do aplicativo). A diferença entre o Titanium e os outros frameworks descritos abaixo é a geração de interfaces que não utilizam de um motor de navegação, mas que são nativas, geradas por código nativo resultante da compilação do código JavaScript (HARTMANN; STEAD; DEGANI, 2011).

O Appcelerator Titanium suporta, a partir do mesmo código fonte, a compilação de aplicativos nativos para iOS, Android, Blackberry, Windows e além disso gera páginas web para outros dispositivos móveis. Além do fato de todas estas plataformas serem contempladas pelo titanium, o fato de ser utilizado o JavaScript como linguagem de desenvolvimento faz com que muitos desenvolvedores adotem o Titanium como SDK de desenvolvimento. Além disso, a utilização de uma ferramenta única permite que seja mais fácil manter os projetos de aplicativos móveis multiplataforma, onde a alteração é única para todas as plataformas (APPCELERATOR, 2013).

Atualmente o Titanium possui mais de 468,667 desenvolvedores contribuindo para o projeto, melhorando os recursos existentes e desenvolvendo novos recursos. Este número é formado por desenvolvedores independentes, parceiros da appcelerator ou funcionários, que desenvolvem novos módulos e extensões para melhorar e facilitar o desenvolvimento das aplicações pelos utilizadores do SDK (APPCELERATOR, 2013).

3.2 Máquina Virtual

Ferramentas do tipo Máquina Virtual ficam entre as ferramentas baseadas em Web e as ferramentas de Compilação Cruzada. Nelas o código-fonte é empacotado juntamente com bibliotecas e uma máquina virtual, que em tempo de execução converte os comandos do código fonte em comandos nativos da plataforma que está executando a aplicação. Sua desvantagem é que normalmente aplicações maiores apresentam um certo delay na sua execução, devido a conversão do código (HARTMANN; STEAD; DEGANI, 2011).

3.2.1 Rhodes

Lançado em 2008, o Rhodes é um framework para desenvolvimento que faz parte do ecossistema de ferramentas voltadas para o desenvolvimento multiplataforma. Por se utilizar de uma Máquina virtual para a execução dos programas desenvolvidos, o mesmo converte as aplicações desenvolvidas utilizando Ruby na sua parte lógica, e *HyperText Markup Language* (HTML), JavaScript e *Cascading Style Sheets* (CSS) para a interface gráfica para código nativo da plataforma em que a aplicação está sendo executada, apresentando uma experiência de usuário similar a de aplicações desenvolvidas em código nativo para a plataforma em que a aplicação está sendo executada (HARTMANN; STEAD; DEGANI, 2011).

3.3 Web

Ferramentas do tipo Web empacotam código fonte voltado para a internet (normalmente desenvolvidos em HTML e CSS) juntamente com um Webkit, dando a impressão para o usuário que se trata de uma aplicação desenvolvida para aquela plataforma. Possui como principal vantagem o desenvolvimento simplificado porém , por não utilizar componentes gráficos do sistema operacional hospedeiro, apenas simula o comportamento de uma aplicação nativa. Além disso, depende dos recursos fornecidos pelo webkit do sistema operacional para determinar o nível de acesso a funcionalidades do mesmo (HARTMANN; STEAD; DEGANI, 2011).

3.3.1 PhoneGap

Criado no inicio de 2008, o PhoneGap fornece um conjunto de ferramentas para desenvolvimento de aplicações multiplataforma para dispositivos móveis utilizando apenas código HTML, JavaScript e CSS. Muito popular pela sua flexibilidade, arquitetura simples e de fácil utilização. Utilizando um modelo híbrido de execução, um único código-fonte escrito em HTML, JavaScript e CSS são executados em um browser empacotado em forma de aplicação nativa, suportado em cada plataforma de destino. O Acesso as funcionalidades do Sistema Operacional é feito por meio da chamada de métodos JavaScript que fazem as requisições na API proprietária do Sistema Operacional. É uma alternativa para a portabilidade de aplicações Web para dispositivos móveis, mas deve-se lembrar que não se possui acesso aos componentes gráficos nativos do sistema operacional (HARTMANN; STEAD; DEGANI, 2011).

4 COMUNICAÇÃO

4.1 Redes Sem Fio

Criadas para substituirem as redes cabeadas em locais onde o cabeamento não é possível, as redes sem fio (do inglês *Wireless*) utilizam frequências de rádio para a transmissão das informações (UFLA, 2009).

As redes sem fio tem como principal característica não haver necessidade de cabos para conectar vários dispositivos. Este tipo de rede apresenta uma complexidade maior comparada com as redes cabeadas, devido a mobilidade dos dispositivos, o que não ocorre com os dispositivos utilizados nas redes com fio, onde a mobilidade máxima é a permitida pelo cabo ao qual o dispositivo está conectado. Além disso, neste tipo de rede podem ocorrer problemas de interferência entre redes que compartilham o mesmo espaço físico (IEEE, 2012).

Apesar das discussões sobre a confiabilidade e a segurança das redes sem fio, existe um consenso de que a configuração fácil, flexibilidade e gerenciamento das redes sem fio fazem com que a mesma sejam uma opção viável quando existe a mobilidade dos dispositivos ou que diferentes dispositivos conectem-se temporariamente a rede. Além disso, graças aos padrões e protocolos é possível que as redes sem fio comuniquem com as redes cabeadas, permitindo a troca de informações entre as mesmas (UFLA, 2009).

4.2 Telefonia e Internet Móvel

Desde a primeira geração de telefones móveis introduzidas no mundo durante os anos 80 e 90 até a atual tecnologia 4G (LTE) muitas evoluções ocorreram na forma como os celulares se comunicam. A evolução causada por estas tecnologias mudou a forma em que o mundo se comunica, trazendo a possibilidade de conectar-se a internet utilizando um dispositivo móvel onde quer que você esteja, bastando haver sinal da operadora de telefonia.

4.2.1 1G

As redes de 1ª Geração (1G) causaram grande impacto na sociedade pelo seu nível de inovação. Trabalhando de forma analógica utilizando-se de modulação de frequência (do inglês Frequency Modulation - FM), onde se transmitia a voz do usuário em faixas de Frequência Muito Alta (do inglês Ultra High Frequency - UHF) (RAMOS, 2012). Este tipo de rede permite apenas o tráfego de voz, onde a qualidade das ligações varia conforme o nível de interferência. Além disso um dos problemas desta tecnologia é a baixa segurança, onde é possível fazer escuta de ligações utilizando-se de sintonizador de rádio e até mesmo utilizar frequências alheias para efetuar ligações (REGO, 2011).

4.2.2 2G

Inserida no mercado no início da década de 90, é marcada pela mudança da tecnologia analógica para a tecnologia digital, permitindo assim além de um maior número de ligações simultâneas na rede, o envio de mensagens de texto (*Short Message Service* (SMS)) e a capacidade de transmitir dados em baixa velocidade entre dispositivos utilizando-se da rede (REGO, 2011).

4.2.3 3G

A terceira geração de redes móveis é considerada um avanço nas redes 2G baseadas na família de normas da União Internacional de Telecomunicações. Trouxe as redes maior capacidade e serviços para os usuários, como conexões de dados a longas distancias com velocidades variando entre 5 e 10Mbps (REGO, 2011).

4.2.4 4G

Absorvendo todas as características das redes 3G, as redes 4G propõe mais velocidade e uma nova visão de mercado. Este padrão totalmente baseado em endereçamento IP garante velocidades de acesso entre 100Mbps com dispositivos em movimento e 5Gbps para dispositivos em repouso. Como o 4G é totalmente baseado em IP, também torna possível a integração de qualquer dispositivo que utilize desta tecnologia, como web TV's, gadgets. A tecnologia prevê a serviços como envio de Mensagens Multimídia (do inglês *Multimedia Messaging Service* -

MMS), Video Chat, TV Móvel, Broadcast de Video Digital além dos serviços básicos como voz e dados. Além disso prevê a a interoperabilidade entre os diversos padrões de rede sem fio (REGO, 2011).

5 SISTEMA ACADÊMICO MINHA UNO

Desenvolvido pela Unochapecó, o sistema acadêmico “Minha Uno” é a interface utilizada pelos acadêmicos, docentes e funcionários da universidade para envio e recebimento de informações.

O sistema consiste em uma página web, dividida em perfis, com opções diferentes para cada um deles. O sistema é composto pelos perfis “Graduação”, “Pós-Graduação”, “Professor”, “Técnico-Administrativo”, “Administrador”, “Fornecedor” e “Apoio-Operacional”.

Dentre os perfis existentes no sistema, foram avaliados os perfis: Professor, Graduação, Pós-Graduação e Técnico-Administrativo.

5.1 Graduação

O perfil de graduação é voltado aos acadêmicos dos diferentes cursos da Unochapecó. Por meio dele, os acadêmicos são informados das suas notas, recebem novos materiais, se informam da sua situação financeira, solicitam documentos entre outras opções.

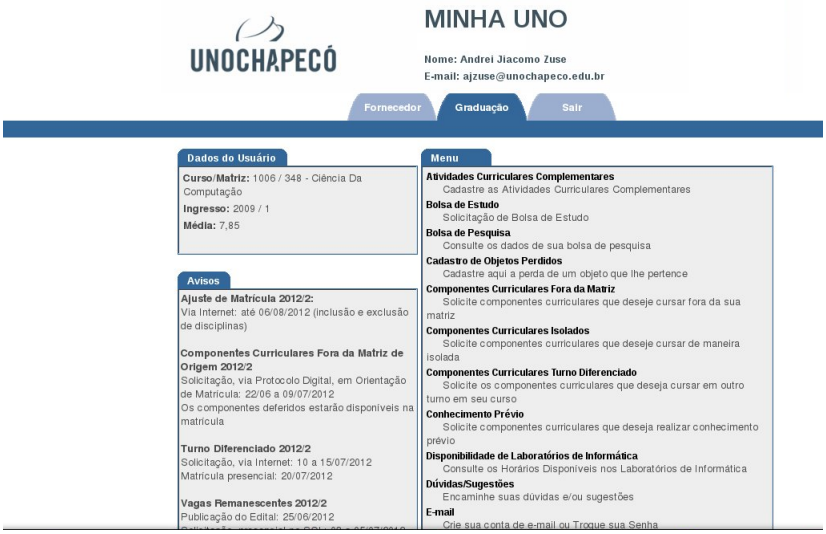
Com uma interface confusa em alguns momentos, porém de modo geral de fácil aprendizado, a mesma é de uso obrigatório para qualquer graduando da instituição, pois sem ela não é possível acompanhar as notas, entregar alguns trabalhos ou tirar os boletos para o pagamento das mensalidades. Abaixo lista completa das opções fornecidas por este perfil:

- Atividades Curriculares Complementares
- Bolsa de Estudo
- Bolsa de Pesquisa
- Cadastro de Objetos Pedidos
- Componentes Curriculares Fora da Matriz
- Componentes Curriculares Isolados

- Componentes Curriculares Turno Diferenciado
- Conhecimento Prévio
- Disponibilidade de Laboratórios de Informática
- Dúvidas/Sugestões
- E-mail
- Entrega de Trabalhos
- Ficha de Matrícula
- Financiamento
- Formulário para Negociação Diferenciada
- Histórico
- Horários de Aula/Ementas/Requisitos
- Horários do Semestre
- Inscrições em Eventos
- Inscrições para Estágio
- Material de Apoio/Planos de Ensino
- Negociação pela Internet
- Notas Graduação
- Pedidos de Livros Livraria Universitária
- Programa de Incentivos
- Protocolo Digital
- Quero Livro - Curso de Direito
- Renovação de Matricula
- Situação Financeira
- Solicitação de Documentos
- Solicitação de Estágio - Cursos de Direito e Farmácia
- Titulos a Receber e
- Trancamento parcial.

O layout do sistema acadêmico e a disposição dos itens pode ser visualizado na Figura 2.

FIGURA 2 - Layout do perfil Graduação do Sistema Acadêmico Minha Uno.



Fonte: Do Autor

5.2 Pós-Graduação

O perfil de pós-graduação é voltado aos acadêmicos dos diferentes cursos de pós-graduação da Unochapecó. Por meio dele, os acadêmicos são informados das suas notas, recebem novos materiais, se informam da sua situação financeira, solicitam documentos entre outras opções.

Com uma interface mais limpa que a apresentada pelo perfil de graduação, a mesma é de uso obrigatório para qualquer pós-graduando da instituição, pois sem ela não é possível acompanhar as notas, entregar alguns trabalhos ou tirar os boletos para o pagamento das mensalidades. As opções presentes neste perfil são:

- Cadastro de Objetos Perdidos
- Dúvidas/Sugestões
- E-mail
- Ementas
- Entrega de Trabalhos
- Inscrições em Eventos
- Material de Apoio/Planosd e Ensino
- Notas Pós-Graduação
- Pedidos de Livros Livraria Universitária
- Programa de Incentivos

- Protocolo Digital
- Situação Financeira
- Solicitação de Documentos e
- Títulos a Receber.

O layout do sistema acadêmico e a disposição dos itens pode ser visualizado na Figura 3.

FIGURA 3 - Layout do perfil Pós-Graduação do Sistema Acadêmico Minha Uno.

Fonte: Do Autor

5.3 Professor

Utilizado pelos docentes da instituição, o perfil de Professor pode ser considerado o painel de controle das disciplinas ministradas. Utilizando-se do perfil no Sistema Acadêmico o professor coloca novos materiais nas disciplinas ministradas, adiciona as notas dos alunos, envia mensagens aos mesmos, efetua as chamadas entre outras funcionalidades. As opções disponíveis neste perfil são:

- Cadastro de Objetos Perdidos
- Cadastro de Veículo
- Componentes Curriculares Complementares
- Comunicação Interna Eletrônica
- Contato RH
- Diário de Classe On-Line

- Documentos Diversos
- E-mail
- Entrega de Trabalhos
- Envio de Projetos de Pesquisa
- Folha de Pagamento
- Gastos dos Convênios Asser
- Horários de Aula/Ementas/Requisitos
- Horários do Professor
- Inscrições em Eventos
- Ligações Telefônicas
- Material de Apoio
- Pedidos de Livros Livraria Universitária
- Período de Férias
- Plano de Ensino
- Plano Mensal de Trabalho do Professor
- Processo Seletivo
- Programa de Aprendizagem
- Quero Livro - Curso de Direito
- Ramais
- Registro das Atividades Mensais
- Relatório on-line/Projetos de pesquisa
- Repositório de Arquivos
- Reservas
- Reservas Laboratório de Informática
- Sistema de Mensagem Integrada
- Solicitação de Coffee Break e
- Sumula de Currículo.

O layout do sistema acadêmico e a disposição dos itens pode ser visualizado na Figura 4.

FIGURA 4 - Layout do perfil Professor do Sistema Acadêmico Minha Uno.

Fonte: Do Autor

5.4 Técnico-Administrativo

Utilizado pelos Funcionários da Unochapecó para comunicação interna, entrar em contato com o setor de Recursos Humanos (RH), consultar sua folha de pagamento, participar de processos seletivos entre outras tarefas, o perfil Técnico-Administrativo possui as seguintes opções:

- Cadastro de Artigos/Monografias/TCC
- Cadastro de Objetos Perdidos
- Cadastro de Veículo
- Cartão-Ponto
- Comunicação Interna Eletrônica
- Contato RH
- Documentos Diversos
- E-mail
- Folha de Pagamento
- Gastos dos Convênios Asser
- Inscrições em Eventos
- Ligações Telefônicas
- Pedidos de Livros Livraria Universitária - Período de Férias
- Processo Seletivo
- Ramais

- Repositório de Arquivos
- Reservas
- Reservas Laboratório de Informática
- Sistema de Mensagem Integrada
- Solicitação de Coffee Break
- Sumula de Currículo e
- Títulos a Receber.

O layout do sistema acadêmico e a disposição dos itens pode ser visualizado na Figura 5.

FIGURA 5 - Layout do perfil Técnico-Administrativo do Sistema Acadêmico Minha Uno.

Fonte: Do Autor

6 QUESTIONÁRIO

Com a finalidade de obter informações sobre a utilização de dispositivos móveis no meio acadêmico, e descobrir a importância de cada item presente no sistema acadêmico atual, entre os dias 25 de agosto de 2012 e 26 de setembro do mesmo ano foi realizado questionário virtual com o corpo acadêmico da universidade.

Ao total 281 pessoas responderam o questionário, sendo obtidas 300 respostas sobre os diferentes perfis do sistema acadêmico. O questionário teve sua estrutura dividida em três partes, sendo a primeira sobre a utilização de smartphones e tablets no meio acadêmico, a segunda parte sobre a relevância de cada item presente no sistema acadêmico atual (Minha Uno) e a terceira parte sobre o interesse de participar dos testes do aplicativo.

Para o questionário, o perfil "Técnico-Administrativo" foi tratado como "Funcionários", sendo esta a nomenclatura adotada na avaliação dos dados coletados.

6.1 Utilização de Dispositivos Móveis

A primeira parte do questionário foi aplicada específicamente para pessoas que possuem dispositivos móveis do tipo smartphone ou tablet (ou ambos), totalizando 126 pessoas (44,84% dos entrevistados).

Com o objetivo de descobrir quais os sistemas operacionais dos dispostivos móveis dos participantes do questionário, foi feita a seguinte pergunta: "Qual o sistema operacional do seu smartphone ou tablet", sendo as alternativas "Android", "iOS", "Windows Phone", "Symbian", "Outros", "Não Sei" e que foi permitido ao usuário marcar mais de uma alternativa. Os resultados obtidos com esta pergunta mostraram que 56,25% dos participantes utilizam aparelhos com o sistema operacional Android, seguidos por 17,97% com iOS, 10,16% com Symbian, 3,91% com Windows Phone e 1,56% com Outros Sistemas Operacionais. Pessoas que não sabiam qual

o sistema operacional do seu smartphone ou tablet totalizaram 10,16%.

Com o objetivo de descobrir qual a forma de acesso a internet móvel mais comum entre os participantes do questionário, foi feita a eles a pergunta "Você utiliza o dispositivo móvel para acessar a internet?" com as opções "Sim, 3G e Wi-fi", "Sim, Apenas 3G", "Sim, Apenas Wi-fi" e "Não", sendo que o participante poderia selecionar apenas uma das opções. Obteve-se que 46,83% dos usuários acessam apenas internet Wi-fi dos seus dispositivos. Em segundo lugar, temos os usuários que acessam internet 3G e Wi-fi, com 45,24% das respostas. Em terceiro lugar aparecem os usuários que não acessam a internet pelos seus dispositivos móveis, com 4,76% e em quarto lugar os usuários que acessam a internet apenas via 3G, com 3,17%.

A última questão feita nesta etapa do questionário se referia ao fato de acessar o sistema acadêmico dos seus dispostivos móveis, sendo que 50,79% dos participantes informaram que não constuma acessar o sistema acadêmico dos seus dispositivos móveis enquanto o restante (49,21%) efetuam este tipo de acesso.

A partir dos dados analisados acima, é possível verificar que, a partir do percentual de participantes que possuem dispositivos móveis, e com os dados do balanço social de 2011 da Fundeste (instituição mantentenedora da Unochapecó), considerando apenas o percentual de pessoas que acessam o sistema acadêmico pelos dispositivos móveis atualmente, aproximadamente 2181 pessoas seriam beneficiadas diretamente com o desenvolvimento de uma aplicação para dispositivos móveis, sendo que se considerar o percentual total de usuários de smartphones ou tablets entrevistados, este número sobe para 4434 pessoas beneficiadas.

6.2 Relevância de cada item presente no sistema acadêmico atual

Com o objetivo de descobrir quais opções disponíveis atualmente no sistema acadêmico são relevantes para os usuários, esta parte do questionário foi dividida em quatro subpartes, onde perguntas específicas sobre cada perfil de acesso do sistema acadêmico foram criadas.

Ao entrar nesta sessão de perguntas, o usuário inicialmente selecionou qual perfil do sistema acadêmico ele faz parte, e posteriormente foi redirecionado as perguntas específicas de cada perfil. Ao fim das perguntas, o mesmo foi direcionado novamente para o formulário de seleção de perfis, para caso possua acesso a mais de um perfil de usuário, poder responder as perguntas referentes aos outros perfis.

Os perfis de interesse do questionário são os perfis referentes aos alunos de Graduação, alunos de Pós-Graduação, Professores e Funcionários, não constando no questionário o perfil Fornecedor ou outros perfis do sistema acadêmico.

Participaram desta etapa do questionário 281 usuários, onde o perfil "Graduação" obteve 280 respostas, o perfil "Pós-Graduação" obteve uma resposta, o perfil "Professor" obteve duas respostas e o perfil "Funcionário" obteve 17 respostas.

6.2.1 Graduação

Esta parte do questionário foi destinada apenas para estudantes de cursos de Graduação, onde o acadêmico deu notas de 1 a 5 para cada item disponível no sistema acadêmico atual, utilizando-se da escala de Likert, onde a nota 1 demonstra que o item menos importante para o acadêmico, e a nota 5 representa que o item é importantíssimo.

Na tabela 1 serão representadas as perguntas efetuadas e a média final das respostas. Esta parte da pesquisa contou com 280 respostas, 93,33% do total de respostas da pesquisa.

TABELA 1 - Média de Respostas dos Alunos de Graduação sobre os itens do sistema acadêmico

Item	**Média Final**
Bolsa de Pesquisa	2,84
Disponibilidade de Laboratório de Informática	2,46
Entrega de Trabalhos	3,70
Histórico	3,79
Horário de Aula/Ementas/Requisitos	4,29
Horários do Semestre	4,39
Material de Apoio/Planos de Ensino	4,80
Notas da Graduação	4,53
Situação Financeira	4,24
Média	**3,8**
Desvio-Padrão	**0,79**

Fonte: elaboração do autor.

Aplicando-se a distribuição t de Student, representada pela fórmula

$$t = \frac{\overline{X} - \mu_0}{\frac{S_c}{\sqrt{n}}}$$

onde:

- $\overline{X}$ é a média amostral observada;

- μ_0 é a média esperada, sendo esta 3 na pesquisa;
- S_c o desvio padrão amostral corrigido, e;
- n é o tamanho da amostra analisada.

O desvio-padrão amostral corrigido é calculado por meio da fórmula

$$S_c = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(X_i - \overline{X})^2}$$

Que apresenta parâmetros semelhantes aos utilizados na fórmula utilizada no cálculo da distribuição t de Student, com o acréscimo do X_i que representa o valor observado para o indivíduo i da amostra (WEISSTEIN, 2012).

Aplicando-se as fórmulas acima a tabela 1, foram obtidos os valores apresentados na tabela 2.

TABELA 2 - Avaliação dos dados obtidos com as respostas dos alunos de graduação

Item	**Média**	S_c	t	**Índice de Confiança (%)**
Bolsa de Pesquisa	2,84	1,435	-1,9158	94,35
Disponibilidade de Laboratório de Informática	2,46	1,351	-6,6773	99,99
Entrega de Trabalhos	3,70	1,350	8,7214	99,99
Histórico	3,79	1,272	10,339	99,99
Horário de Aula/Ementas/Requisitos	4,29	1,067	20,221	99,99
Horários do Semestre	4,39	0,906	25,738	99,99
Material de Apoio/Planos de Ensino	4,80	0,553	54,334	99,99
Notas da Graduação	4,53	0,552	46,475	99,99
Situação Financeira	4,24	0,903	22,899	99,99

Fonte: elaboração do autor.

O cálculo do índice de confiança pode ser reproduzido utilizando-se da função INVT (ou similar) do editor de planilhas eletrônicas, sendo que os percentuais apresentados no índice de confiança foram arredondados para duas casas decimais para permitir melhor interpretação do mesmos. Devido a utilização da distribuição t de Student na forma bicaudal, nos casos onde o valor t apresentou-se negativo, o índice de confiança representa o percentual de chances de a resposta ficar abaixo da média em pesquisas feitas posteriormente, enquanto em itens onde o valor t apresentou-se positivo, o percentual de confiança demonstra as chances de em pesquisas posteriores os valores ficarem acima da média.

6.2.2 Pós-Graduação

Esta parte do questionário foi destinada apenas para estudantes de cursos de Pós-Graduação, onde o pós-graduando deu notas de 1 a 5 para cada item disponível no sistema acadêmico atual, utilizando-se da escala de Likert, onde a nota 1 demonstra que o item menos importante para o acadêmico, e a nota 5 representa que o item é importantíssimo.

Na tabela 3 serão representadas as perguntas efetuadas e a média final das respostas. Esta parte da pesquisa contou com 1 resposta, não sendo assim possível extrair informações conclusivas sobre este perfil do sistema acadêmico.

TABELA 3 - Média de Respostas dos Alunos de Pós-Graduação sobre os itens do sistema acadêmico

Item	**Média Final**
Ementas	3,00
Entrega de Trabalhos	5,00
Material de Apoio/Planos de Ensino	5,00
Notas da Pós-Graduação	5,00
Situação Financeira	5,00
Média	**4,60**
Desvio-Padrão	**0,89**

Fonte: elaboração do autor.

Devido a possuir apenas uma resposta, não foi efetuada avaliação de Student sobre este item para medir o índice de confiança da resposta obtida, pois o tamanho da amostra pode ser considerada insignificante para a tomada de decisões.

6.2.3 Professor

Esta parte do questionário foi destinada apenas para o corpo docente da instituição onde o professor deu notas de 1 a 5 para cada item disponível no sistema acadêmico atual, utilizando-se da escala de Likert, onde a nota 1 demonstra que o item menos importante para o acadêmico, e a nota 5 representa que o item é importantíssimo.

Na tabela 4 serão representadas as perguntas efetuadas e a média final das respostas. Esta parte da pesquisa contou com 2 respostas, não sendo assim possível extrair informações conclusivas sobre este perfil do sistema acadêmico.

TABELA 4 - Média de Respostas dos Professores sobre os itens do sistema acadêmico

Item	**Média Final**
Componentes Curriculares Ministrados	1,50
Diário de Classe Online	5,00
Documentos Diversos	1,00
Entrega de Trabalhos	4,50
Folha de Pagamento	2,00
Gastos dos Convênios Asser	1,50
Horários de Aula/Ementas/Requisitos	4,50
Horários do Professor	4,00
Ligações Telefônicas	2,00
Material de Apoio	5,00
Período de Férias	1,00
Plano de Ensino	4,00
Processo Seletivo	2,00
Programa de Aprendizagem	3,00
Ramais	3,00
Registro de Atividades Mensais	2,50
Sistema de Mensagens Integrada	4,00
Média	**2,97**
Desvio-Padrão	**1,40**

Fonte: elaboração do autor.

Além das questões referentes aos itens acima, também foi feita a pergunta *Sobre o item "Diário de Classe Online", seria interessante a possibilidade de fazer chamadas e registrar notas pelo smartphone ou tablet?* sendo que 100% dos participantes responderam afirmativamente.

Devido a possuir apenas duas respostas, não foi efetuada avaliação de Student sobre este item para medir o índice de confiança das respostas obtidas, pois o tamanho da amostra pode ser considerada insignificante para a tomada de decisões.

6.2.4 Funcionário

Esta parte do questionário foi destinada apenas para os funcionários da instituição onde o funcionário deu notas de 1 a 5 para cada item disponível no sistema acadêmico atual, utilizando-se da escala de Likert, onde a nota 1 demonstra que o item menos importante para o acadêmico, e a nota 5 representa que o item é importantíssimo.

Na tabela 5 serão representadas as perguntas efetuadas e a média final das respostas.

Esta parte da pesquisa contou com 17 respostas.

TABELA 5 - Média de Respostas dos Funcionários sobre os itens do sistema acadêmico

Item	**Média Final**
Cartão-Ponto	4,29
Folha de Pagamento	4,59
Gastos dos Convênios Asser	2,24
Ligações Telefônicas	3,00
Período de Férias	3,41
Processo Seletivo	3,35
Ramais	3,65
Sistema de Mensagem Integrada	3,59
Súmula de Currículo	3,12
Títulos a Receber	3,41
Média	**3,46**
Desvio-Padrão	**0,66**

Fonte: elaboração do autor.

Aplicando-se a distribuição t de Student, explicada na seção 7.2.1, foram obtidos os índices de confiança mostrados na tabela 6.

TABELA 6 - Avaliação dos dados obtidos com as respostas dos funcionários da instituição

Item	**Média**	S_c	t	**Índice de Confiança (%)**
Cartão-Ponto	4,29	0,314	16,986	99,99
Folha de Pagamento	4,59	0,190	34,388	99,99
Gastos dos Convênios Asser	2,24	0,344	-9,4499	99,99
Ligações Telefônicas	3,00	0,339	0	0
Período de Férias	3,41	0,306	5,5489	99,99
Processo Seletivo	3,35	0,293	4,9738	99,98
Ramais	3,65	0,359	7,4393	99,99
Sistema de Mensagem Integrada	3,59	0,306	7,927	99,99
Súmula de Currículo	3,12	0,292	1,662	88,5
Títulos a Receber	3,41	0,282	6,0298	99,99

Fonte: elaboração do autor.

O item “Ligações Telefônicas” apresentou indice de confiança igual a 0 pois a média obtida no mesmo é igual a média esperada (μ_0) para o questionário, tornando a parte superior da fórmula apresentada na seção 7.2.1 zero. Sendo assim a média a ser obtida em questionários posteriores pode ser superior ou inferior a obtida neste questionário.

Para os outros itens aplicam-se as regras presentes na seção 7.2.1, o cálculo do índice de confiança pode ser reproduzido utilizando-se de um editor de planilhas eletrônicas, sendo

que neste trabalho os percentuais apresentados no índice de confiança foram arredondados para duas casas decimais para permitir melhor interpretação do mesmos.

Devido a utilização da distribuição t de Student na forma bicaudal, nos casos onde o valor t apresentou-se negativo, o indice de confiança representa o percentual de chances de a resposta ficar abaixo da média em pesquisas feitas posteriormente, enquanto em itens onde o valor t apresentou-se positivo, o percentual de confiança demonstra as chances de em pesquisas posteriores os valores ficarem acima da média.

6.3 Avaliação dos dados coletados

Avaliando-se os resultados obtidos no questionário, dados estes apresentados anteriormente, observa-se que os perfis utilizados pelos professores e alunos de pós-graduação não possuem informações suficientes para se chegar a conclusões sobre os itens importantes ou não do sistema. Por outro lado, os perfis utilizados pelos funcionários e estudantes de graduação obtiveram uma quantidade maior de respostas, fornecendo assim dados mais conclusivos sobre a relevancia dos itens presentes nestes perfis.

Conclui-se que para a implementação do aplicativo para dispositivos móveis, deve-se observar os itens que obtiveram maior relevância nestes perfis em que foi possível efetuar uma análise mais detalhada, onde os itens que obtiveram maiores médias terão maior prioridade sobre os itens que obtiveram médias menores. Além disso, os itens que ficaram com suas médias abaixo de 3,00 não serão considerados importantes na etapa inicial de desenvolvimento, sendo implementados apenas caso haja tempo suficiente após os itens com maiores médias serem implementados no aplicativo.

6.4 Interesse em participar dos testes da nova aplicação

Contando apenas com duas perguntas, a etapa final do questionário teve como objetivos levantar interessados em auxiliar nos testes da aplicação, e obter o contato das pessoas interessadas. Tendo como base os pesquisados que possuem smartphone ou tablet, 99,21% demonstraram interesse em auxiliar nos testes da nova aplicação.

6.5 Implementação

Com base nos dados apresentados neste capítulo, optou-se em serem implementado apenas o perfil dos alunos de graduação, devido ao número de respostas trazer informações mais confiáveis sobre as opções mais utilizadas pelos acadêmicos. Além disso, devido a forma de extração das informações ocorrer de forma manual e depender do layout da página do sistema acadêmico atual, foi optado por implementar os três módulos com melhores médias (Material de Apoio/Planos de Ension, Notas da Graduação e Horários do Semestre), para demonstrar a possibilidade da extração e também possuir informações reais para a aplicação móvel, extraidas diretamente do Sistema Acadêmico Minha Uno.

7 SERVIDOR

Com o objetivo de extrair e preparar as informações a serem consumidas pela aplicação móvel do sistema acadêmico Minha Uno, foi desenvolvido em linguagem Java um servidor REST para facilitar e agilizar a extração dos dados.

Como no questionário apresentado anteriormente obtemos maior quantidade de respostas referentes aos acadêmicos de cursos de graduação, então para este trabalho foi escolhido implementar o servidor e posteriormente a aplicação para este público alvo, sendo implementados os módulos que apresentaram as três melhores notas, sendo eles Material de Apoio, Notas da Graduação e Horários do Semestre.

Utilizando-se da biblioteca *jsoup* para a extração das informações e após o tratamento dos dados gerando-se arquivos JSON transmitidos utilizando um servidor REST, serão detalhadas nas próximas sessões o funcionamento de cada etapa da extração, preparação e transmissão das informações desde o sistema acadêmico Minha Uno até a aplicação móvel.

7.1 Ferramentas Utilizadas

Para o desenvolvimento do servidor foram utilizadas apenas ferramentas Open-Source, sendo que a linguagem escolhida para o desenvolvimento foi o Java, devido a quantidade de documentação encontrada na internet, e também por possuir bibliotecas prontas que permitem a extração, tratamento e disponibilidade das informações, facilitando assim a implementação do servidor.

7.2 Extração das Informações

Segundo informações obtidas da diretoria de Ti da Unochapecó, a instituição não possui um webservice com as informações do sistema acadêmico, e desta forma as informações

exibidas na página são extraidas diretamente de uma coleção de banco de dados, o que tornaria inviável a integração direta com estas bases. Com a excasses de alternativas, foi necessário extrair as informações diretamente da página do sistema acadêmico. Para todos os processos de extração foi utilizada a biblioteca *jsoup*, sendo que a mesma é responsável pela conexão aos diferentes endereços do sistema acadêmico e pela extração das informações a partir do retorno das consultas de navegação, ou seja, sendo extraidas as informações a partir do HTML retornado pelo sistema acadêmico atual.

Após o devido tratamento utilizando-se filtros com expressão regular e outros comandos permitidos pelo *jsoup*, as informações necessárias para a aplicação são obtidas, sendo que na continuidade do capítulo será explicado como cada grupo de informações foi extraido e posteriormente preparado para ser consumido pelos dispositivos móveis.

Todas as expressões utilizadas na biblioteca *jsoup* podem ser testadas na página web `http://try.jsoup.org/` utilizando como entrada o código HTML da página a ser analizada. Mais informações sobre as expressões utilizadas podem ser obtidas em `http://jsoup.org/cookbook/extracting-data/selector-syntax`.

Como toda a extração é baseada no retorno do código HTML do sistema atual, qualquer alteração no layout do arquivo HTML retornado pela página do sistema acadêmico Minha Uno resultará em problemas na extração dos dados.

7.2.1 Login

Com o objetivo de validar se o login fornecido pelo usuário na aplicação é válido e extrair o cookie (identificador de sessão utilizado para as acesso as informações obtidas apenas pelo login válido do usuário), o servidor possuí como primeira tarefa ao receber uma solicitação da aplicação a validação de login.

Utilizando-se do método POST do HTTP, os dados de login são enviados utilizando o endereço `https://www.unochapeco.edu.br/usuarios/login?login_submited=1&usuario=USUARIO&senha=SENHA&submit=entrar` (as palavras USUARIO e SENHA devem ser substituidas pelas respectivas informações).

O algoritmo de extração do cookie e validação do login não apresenta nenhuma complexidade, conforme pode-se observar no Apêndice A.1 deste trabalho.

7.2.2 Material de Apoio

Com o objetivo de extrair a relação dos materiais eletrônicos postados pelos professores em cada disciplina cursada pelo acadêmico. Para melhor entendimento, o algoritmo será dividido em duas partes, sendo que a primeira parte mostrará como são extraidas as informações referentes as disciplinas cursadas pelo acadêmico, e a segunda parte do como são extraidas as informações dos materiais disponíveis. O código-fonte desenvolvido pode ser conferido no Apêndice A.2 deste trabalho.

7.2.2.1 Extração das Disciplinas

Com o objetivo de extrair a lista das disciplinas pertencentes ao módulo de material de apoio do sistema acadêmico, é necessário que a informação seja extraida a partir do código HTML retornado pela url `https://www.unochapeco.edu.br/saa/materialApoio.php`. Para a obtenção das disciplinas corretas cursadas pelo acadêmico, é utilizado o cookie de sessão capturado no momento do login, conforme explicado no item 5.2.1 deste trabalho. Um exemplo de código HTML retornado pelo servidor a partir deste tipo de consulta pode ser visto na url `http://goo.gl/fLo0h`.

A partir do código HTML retornado, sendo aplicanda a expressão "*form tr td:eq(1) a*" sob o mesmo por meio da biblioteca *jsoup*, obtém-se como retorno a lista das disciplinas cursadas pelo acadêmico, conforme pode ser visto na figura 6.

FIGURA 6 - Lista das disciplinas extraidas.

Fonte: Do Autor.

A partir da lista retornada acima, é possível retornar diretamente a disciplina no formato "*Código - Nome*". Também como pode-se verificar na Imagem 5, acima do nome da disciplina é exibida a tag HTML completa, sendo que nesta tag pode-se verificar o atributo *href*, que possui o caminho completo para serem extraidos os materiais da disciplina.

Após a extração do nome da disciplina e da tag *href*, é necessária a extração das informações dos materiais da disciplina, explicada na sessão 5.2.2.2 deste trabalho.

7.2.2.2 Extração dos Materiais

Após a extração da lista de disciplinas e da referência ao endereço web onde as disciplinas podem ser acessadas, é necessário extrair as informações dos materiais propriamente ditos.

A partir do código HTML (disponível na url `http://goo.gl/jB8ia`, sendo que o mesmo é pertencente a uma disciplina do 9º período de Ciência da Computação na grade 348) obtido pela url capturada (conforme explicado na sessão 5.2.2.1). No código retornado, existem 4 informações relevantes, sendo elas: nome, url, publicação e descrição.

Para a extração do nome do material, a expressão "*form tr:contains(Arquivo) a*", sendo que a mesma retorna uma lista com o nome das disciplinas, conforme pode ser visto na Figura 7.

FIGURA 7 - Lista de nomes dos materiais de uma disciplina.

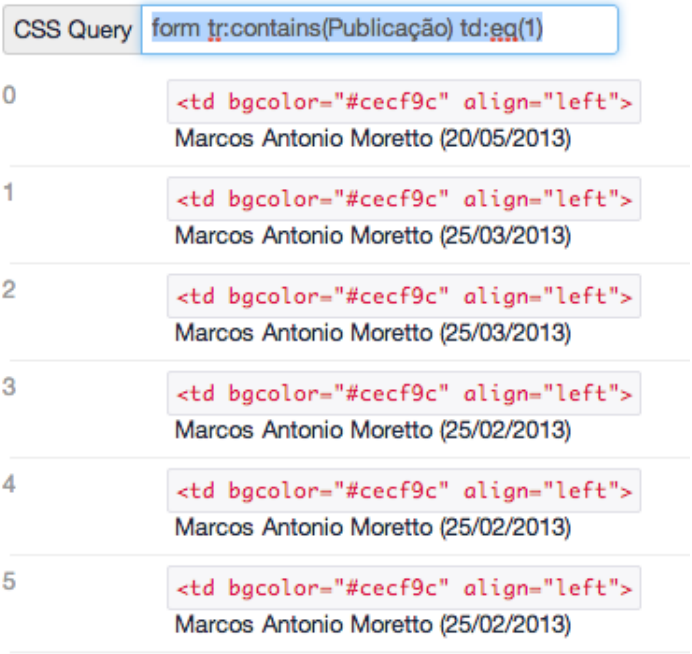
Fonte: Do Autor.

Verificamos na Figura 6 que a mesma expressão retorna no campo principal o nome das disciplinas, e na sua referência (exibida em vermelho) o campo href, que nos é interessante para

o acesso direto ao arquivo listado. Desta forma, utilizando-se da mesma expressão é possível extrair a url de acesso direto ao arquivo e também o nome do arquivo a ser acessado. Desta forma, duas informações já são preenchidas a partir da mesma consulta.

Para obtermos a publicação, que nada mais é o nome do professor que postou o material e a data de postagem, utiliza-se a expressão "*form tr:contains(Publicação) td:eq(1)*", e o seu retorno pode ser verificado na Figura 8.

FIGURA 8 - Lista de publicação dos materiais de uma disciplina.



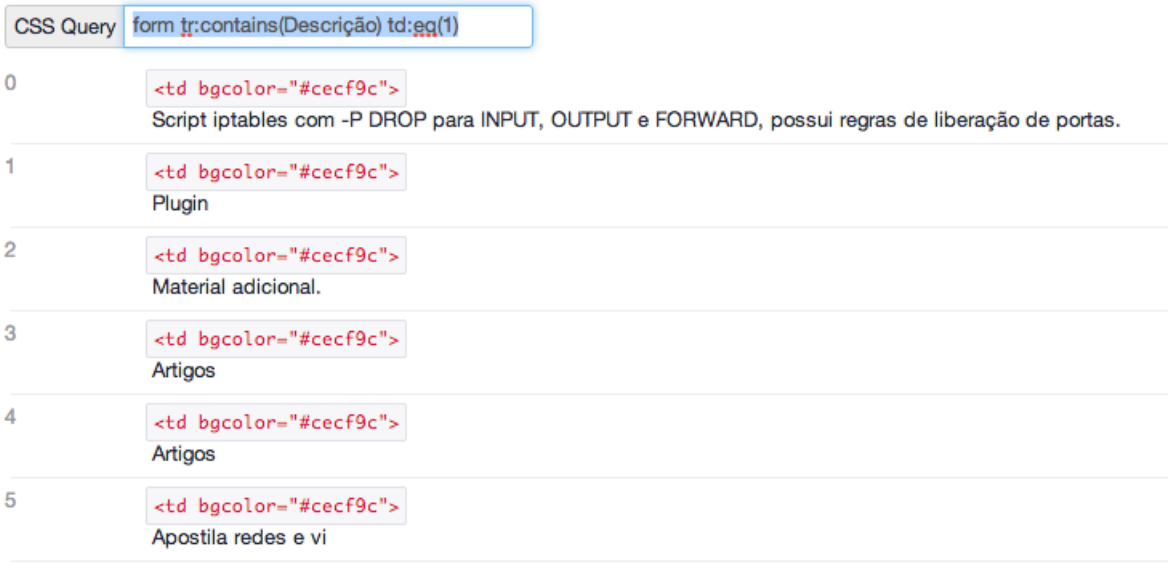
Fonte: Do Autor.

Para obtermos a descrição dos materiais postados, a expressão "*form tr:contains (Descrição) td:eq(1)*" onde obtém-se como resultado a lista da descrições dos materiais, confome pode ser visto na Figura 9.

7.2.3 Notas da Graduação

Com o objetivo de de extrair as avaliações e as respectivas notas das avaliações aplicadas aos acadêmicos, foi desenvolvida a classe de extração das notas da graduação. Para melhor entendimento, o algoritmo será dividido em três partes, onde a primeira representará a extração da lista de disciplinas cursadas pelos acadêmicos, a segunda as informações referentes as avaliações e notas das disciplinas em aberto e a terceira parte explicará a extração das disciplinas já finalizadas pelo acadêmico.

FIGURA 9 - Lista de descrição dos materiais de uma disciplina.



Fonte: Do Autor.

Assim como na sessão 5.2.2, será utilizada a biblioteca *jsoup* para extração das informações, utilizando-se das mesmas recomendações da sessão mencionada anteriormente.

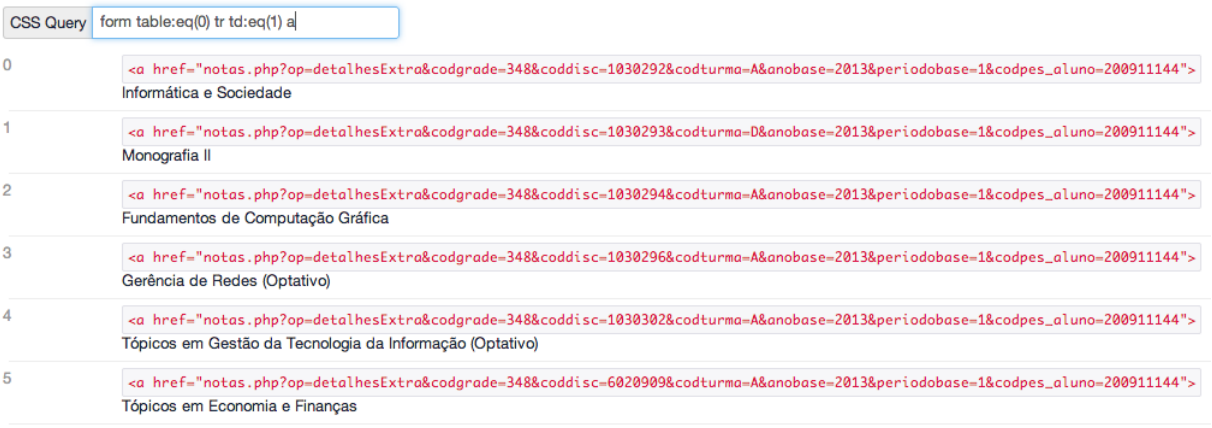
7.2.3.1 Extração das Disciplinas

Com o objetivo de extrair a lista das disciplinas pertencentes ao módulo de notas da graduação do sistema acadêmico, é necessário que a informação seja extraida a partir do código HTML retornado pela url `https://www.unochapeco.edu.br/saa/notas.php`. Para a obtenção das disciplinas corretas cursadas pelo acadêmico, é utilizado o cookie de sessão capturado no momento do login, conforme explicado no item 5.2.1 deste trabalho. Um exemplo de código HTML retornado pelo servidor a partir deste tipo de consulta pode ser visto na url `http://goo.gl/TrrOl`.

A partir do código HTML retornado, sendo aplicanda a expressão "*form table:eq(0) tr td:eq(1) a*" sob o mesmo por meio da biblioteca *jsoup*, obtém-se como retorno a lista das disciplinas cursadas pelo acadêmico, conforme pode ser visto na figura 10.

Como pode ser percebido na Figura 9, além do nome das disciplinas, na parte em vermelho que representa a tag HTML possui o atributo *href*, sendo que o mesmo armazena o link para acesso as avaliações e notas da disciplina em específico, sendo este atributo importante para a a extração das notas e avaliações das disciplinas em aberto, sendo então necessário coletar

FIGURA 10 - Lista das disciplinas das Notas da Graduação.



Fonte: Do Autor.

o mesmo juntamente com o nome da disciplina. Após a extração do nome das disciplinas e da respectiva url a partir do atributo *href*, é iniciada a extração das avaliações por disciplina.

7.2.3.2 Extração das Notas e Avaliações das Disciplinas em Aberto

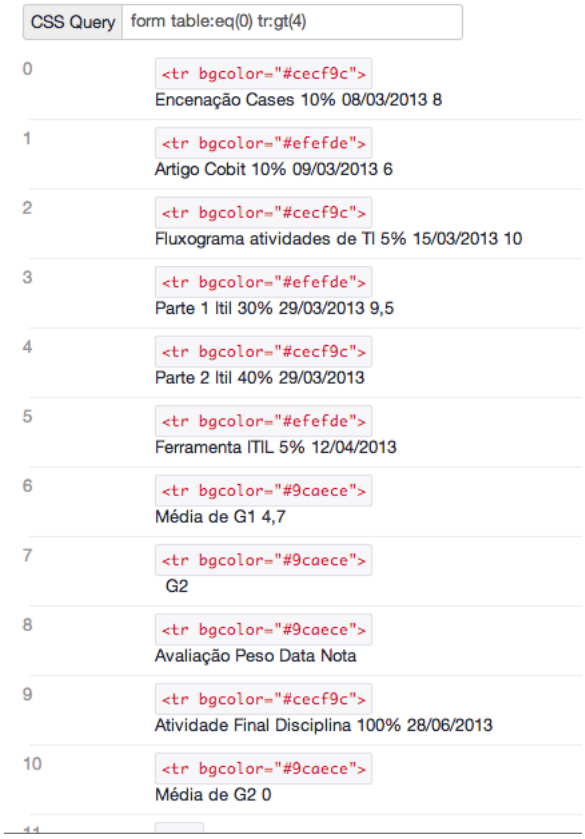
Após a extração do nome das disciplinas cursadas pelo acadêmico e da url utilizada para acesso a disciplina específica, esta mesma url é utilizada para a extração das avaliações e notas do acadêmico, assim como também das médias do graduando. Para esta extração são utilizadas várias combinações de expressões. Um exemplo de código retornado pode ser consultado na url `http://goo.gl/lfTy3`.

Para a extração da lista de avaliações, é utilizada a expressão *form table:eq(0) tr:gt(4)* como base da extração, e posteriormente são aplicadas novas expressões sobre os resultados obtidos da aplicação da expressão base. A Figura 11 nos mostra parcialmente o retorno da expressão.

Como pode ser percebido na Figura 10, as informações aparecem um pouco misturadas, sendo necessário tratar via programação as informações que realmente serão extraidas e ignorando as informações que não são necessárias. Isto é feito por meio de uma verificação de quando a informação contida na quarta coluna da tabela retornada esta em branco ou contém a palavra notas.

A partir do retorno acima, são extraidas as seguintes informações das avaliações: nome, peso, data e nota. Cada uma destas informações possui uma expressão aplicada sobre a expressão base. A Tabela 7 contém a expressão para cada informação e a ser extraida.

FIGURA 11 - Resultado parcial da Expressão base sobre as avaliações.



Fonte: Do Autor.

TABELA 7 - Tabela de expressões de extração das informações das avaliações

Informação	**Expressão**
Nome	td:eq(0)
Peso	td:eq(1)
Data	td:eq(2)
Nota	td:eq(3)

Fonte: elaboração do autor.

Para a extração das médias de G1 e de G2 dos graduandos, são utilizadas as expressões contidas na Tabela 8, sendo que estas expressões são aplicadas diretamente sobre o código HTML, não mais sobre a expressão base.

TABELA 8 - Tabela de expressões de extração das médias de G1 e de G2

Informação	**Expressão**
Média de G1	form table:eq(0) tr:contains(Média de G1) td:eq(1)
Média de G2	form table:eq(0) tr:contains(Média de G2) td:eq(1)

Fonte: elaboração do autor.

Após estes procedimentos a extração das avaliações, assim como das médias de G1 e

G2 estão concluídas, faltando apenas a extração das notas para as disciplinas já finalizadas pelo acadêmico no semestre.

7.2.3.3 Extração das Notas das Disciplilas Finalizadas

Finalizando a sessão de extração das notas da graduação, a extração das notas das disciplinas já finalizadas ocorre em cima do mesmo código HTML obtido no item 5.2.3.1 deste trabalho, que pode ser consultado na url `http://goo.gl/X2xJg`.

Devido a ordem em que as disciplinas são exibididas na lista de disciplinas ser diferente a ordem exibida nas notas oficiais da graduação, a expressão utilizada para a busca das notas de disciplinas já finalizadas leva em consideração o código da disciplina encerrada, extraindo o mesmo a partir do atributo href da tag html retornada na lista de disciplinas. Para o retorno das informações da disciplina já finalizada, é utilizada a expressão *form[name$=graduacao] tr:contains(CODIGODISCIPLINA)*, sendo que a palavra CODIGODISCIPLINA deve ser substituida pelo código propriamente dito.

7.2.4 Horários do Semestre

Com o objetivo de extrair o horário das disciplinas cursadas pelo acadêmico, e também as informações detalhadas da disciplina, foi desenvolvida a classe de extração dos horários do semestre. Para facilitar o entendimento, a explicação do processo será dividido em três partes, sendo elas: Extração do nome das disciplinas, Extração das informações gerais da disciplina e Extração dos Horários de Aula.

7.2.4.1 Extração das Disciplinas

Com o objetivo de extrair a lista das disciplinas pertencentes ao módulo de horários do semestre do sistema acadêmico, é necessário que a informação seja extraida a partir do código HTML retornado pela url `https://www.unochapeco.edu.br/saa/hor_aluno.php`. Para a obtenção das disciplinas corretas cursadas pelo acadêmico, é utilizado o cookie de sessão capturado no momento do login, conforme explicado no item 5.2.1 deste trabalho. Um exemplo de código HTML retornado pelo servidor a partir deste tipo de consulta pode ser visto na url `http://goo.gl/kz9NN`.

A partir do código HTML retornado, utilizando-se das expressões exibidas na Tabela 9 são extraidas as informações de código, nome e turma da disciplina.

TABELA 9 - Tabela de expressões de extração dos horários do semestre.

Informação	Expressão
Código	form tr:gt(1) td:eq(0) a
Nome	form tr:gt(1) td:eq(1) a
Turma	form tr:gt(1) td:eq(2) a

Fonte: elaboração do autor.

Como ocorre na extrações anteriores, é necessário extrair da tag HTML retornada o atributo *href* para serem extraidas tanto as informações gerais como os horários de aula da disciplina.

Após terminada a extração das disciplinas, e possuindo a url armazenada no atributo *href*, agora são extraidas as informações gerais das disciplinas e também os horários de aula.

7.2.4.2 Extração das Informações Gerais

A extração das Informações gerais ocorre a partir da url extraida da disciplina como visto no item 5.2.4.1 deste trabalho. Para a extração das informarmações é necessária uma expressão para cada informação, sendo demonstradas as expressões na tabela 10. O código HTML retornado pela página do sistema acadêmico para uma disciplina de exemplo pode ser consultado na url `http://goo.gl/auHL8`.

TABELA 10 - Tabela de expressões de extração dos detalhes das disciplinas.

Informação	Expressão
Curso	form tr[bgcolor]:contains(curso) td:eq(1)
Grade	form tr[bgcolor]:contains(grade) td:eq(1)
Disciplina	form tr[bgcolor]:contains(disciplina) td:eq(1)
Período	form tr[bgcolor]:contains(período) td:eq(1)
Professor	form tr[bgcolor]:contains(professor) td:eq(1)
Turno	form tr[bgcolor]:contains(turno) td:eq(1)
Créditos	form tr[bgcolor]:contains(créditos) td:eq(1)
Data da G2	form tr[bgcolor]:contains(g2) td:eq(1)
Data da G3	form tr[bgcolor]:contains(g3) td:eq(1)

Fonte: elaboração do autor.

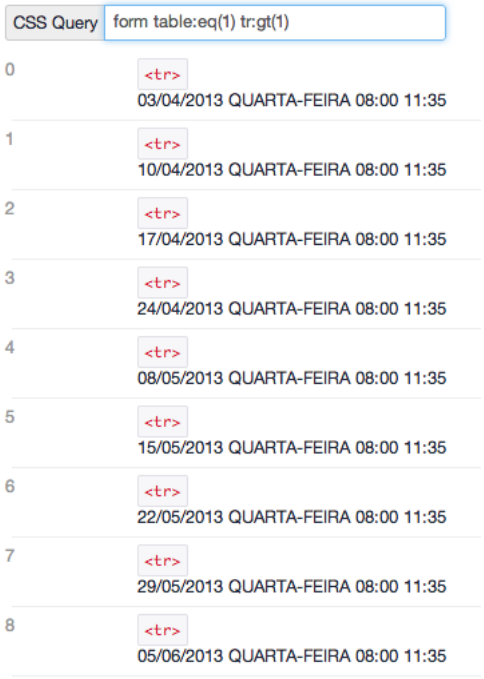
A partir de cada uma das expressões presentes na Tabela 10, obtemos cada uma das informações que fazem parte dos detalhes da disciplina cursada pelo acadêmico, e desta forma

encerramos a extração dos detalhes, iniciando a extração dos horários de aula explicada no item 5.2.4.3.

7.2.4.3 Extração dos Horários de Aula

Como última parte da extração das informações, falaremos da Extração dos horários de aula. Tendo como base a mesma url e código HTML utilizado no item 5.2.4.2 deste trabalho, podemos obter os horários de aula das disciplinas. Para isso é aplicada a expressão *form table:eq(1) tr:gt(1)* que nos retorna a lista com as aulas da disciplina, conforme pode ser visto na Figura 12.

FIGURA 12 - Lista de horários para uma disciplina.



Fonte: Do Autor.

A partir do retorno da expressão demonstrado na Figura 11, são aplicadas novas expressões para obter-se as informações específicas necessárias, sendo estas expressões mostradas na Tabela 11.

Para saber se a aula já foi ministrada ou não, é validado se a informação sendo extraida está em negrito no código HTML, acrescentando-se a expressão *b* a expressão original.

Desta forma, toda a extração de informações utilizando a biblioteca *jsoup* é finalizada, sendo necessário agora preprar o arquivo que será enviado ao aplicativo móvel utilizando o servidor REST.

TABELA 11 - Tabela de expressões de extração dos detalhes das disciplinas.

Informação	Expressão
Data	td:eq(0)
Dia da Semana	td:eq(1)
Hora	td:eq(2)
Ocorreu	b

Fonte: elaboração do autor.

7.3 Preparação dos Dados

A preparação dos dados para o envio para a aplicação móvel é feita em conjunto com a extração dos dados da página, porém para demonstrar de forma mais simples e facilitar o entendimento este capítulo irá demonstrar apenas o layout dos arquivos gerados.

Toda a geração dos arquivos funciona como retorno das funções de extração, sendo que o retorno ocorre em formato JSON, devido ao formato simples de representação de dados e economia da banda de dados se comparado com o formato XML, já que o JSON não possui tags de fechamento e cabeçalho nos arquivos. Não existe muitas coisas a serem explicadas sobre os arquivos gerados, pois os mesmos são de fácil leitura e entendimento, e juntamente com as informações já comentadas no item 5.2 deste trabalho causaria redundância nas informações.

7.4 Servidor REST

Após extração e preparação dos dados, resta apenas disponibilizar as informações para acesso externo. Para esta tarefa foi utilizada a classe Icebreak REST Server (LIISBERG, 2010), que implementa um servidor REST básico que atende as necessidades deste trabalho. Foi necessária apenas uma alteração no cabeçalho HTML retornado pela classe, o qual retornava tamanhos errados, que acabavam ocasionando a perda de informações do arquivo JSON. Como não é um parâmetro obrigatório do cabeçalho do HTML, o tamanho deixou de ser enviado no mesmo, e assim o problema deixou de existir (o código-fonte da implementação da classe com a alteração do cabeçalho pode ser conferida no Apêndice C).

Como todo serviço de rede, é necessário escutar uma porta para receber as conexões, e neste trabalho foi escolhida a porta 90 para o servidor REST, pois assim não ocasionaria um conflito com a porta 80 utilizada pelos servidores HTTP.

Como possuimos três tipos de retorno no servidor, cada um referente a um módulo do sistema acadêmico, para otimizar o envio e também otimizar a extração no lado da aplicação, foram utilizados parâmetros na URL utilizada para conectar ao servidor, sendo estes parâmetros demonstradas na tabela 12.

TABELA 12 - Tabela com os parâmetros aceitos pelo servidor REST.

Parâmetro	**Descrição**
usuario	Neste parâmetro é possível passar ao servidor tanto a matrícula do acadêmico como o seu email da Unochapecó. Parâmetro Obrigatório
senha	Neste parâmetro deve ser informada a senha do acadêmico referente a matrícula ou email informado. Parâmetro Obrigatório.
info	Neste campo é passado o tipo de informação que deseja ser recebida do servidor.

Fonte: elaboração do autor.

O parâmetro *info* presente na Tabela 12 merece uma atenção especial, pois o mesmo não é obrigatório. Caso o mesmo seja omitido da URL de conexão com o servidor, teremos como retorno um valor booleano informando se o login é válido. Nesta implementação do servidor, são três as possíveis opções para o parâmetro *info*: materiais, notas e horarios. Cada uma das opções refere-se a um dos módulos do sistema acadêmico extraidos anteriormente.

O acesso ao servidor ocorre por meio de requisições HTTP na porta 90. Por exemplo, caso se deseje se obter as informações do material de apoio de determinado acadêmico com o nome de usuário *teste* e a senha *teste*, e o servidor estivesse rodando localmente na máquina utilizada para consulta, bastaria utilizar a URL `http://localhost:90?usuario=teste&senha=teste&info=materiais`, sendo assim o servidor encarregado de validar o login e retornar as informações corretas referentes aos materiais de apoio do acadêmico.

8 APLICAÇÃO

Tendo como base o questionário apresentado anteriormente onde obtemos maior quantidade de respostas referentes aos acadêmicos de cursos de graduação, foi decidido para este trabalho implementar a aplicação para este público alvo, sendo implementados os módulos que apresentaram as três melhores notas, sendo eles Material de Apoio, Notas da Graduação e Horários do Semestre.

Para o desenvolvimento da aplicação, foi utilizado exclusivamente a ferramenta de desenvolvimento *Titanium SDK* desenvolvida pela *Appcelerator Inc.* devido ao fato de a mesma ser gratuita e utilizar uma linguagem com curva de aprendizado suave. Além disso o fato de utilizar a mesma e desenvolver em JavaScript utilizando ela facilitou a comunicação com o servidor e a extração dos arquivos JSON retornados por ele. Além disso, utilizando-se o *Titanium SDK* para o desenvolvimento, é possível compilar com o mesmo código fonte aplicações para Android e iOS, plataformas mais utilizadas pelos acadêmicos da instituição. Apesar de ser fortemente lembrados os padrões, nem todo o sistema foi desenvolvido utilizando-se os conceitos de orientação a objetos, sendo utilizadas tanto técnicas de programação estruturada como orientação a objetos conforme a técnica que melhor atendia o desenvolvimento do módulo em específico.

Todas as informações extraidas do servidor são inseridas em uma base de dados SQLite no próprio dispositivo, possibilitando assim a consulta offline das informações. A sincronização com o servidor pode ser feita utilizando-se de rede sem fio na forma de wireless ou utilizando-se de rede de telefonia celular, não havendo restrição na forma de sincronização dos dados.

Para facilitar e modularizar este capítulo, as informações referentes a cada módulo foram separadas em sessões que serão apresentadas a seguir.

8.1 Base de Dados da Aplicação

Com o objetivo de armazenar as informações enviadas pelo servidor, foi modelada uma pequena base de dados composta por 5 tabelas utilizadas no armazenamento de dados e uma tabela para armazenamento das configurações. Para esta base de dados foi utilizado o SQLite, nativo nos dispositivos Android e iOS.

8.1.1 Tabela de Configuração

A tabela de configuração é a única tabela não transmitida pelo servidor para a aplicação, sendo gerada e manipulada no próprio dispositivo. A lista de campos e seus respectivos tipos de dados é demonstrada na Tabela 13.

TABELA 13 - Layout da tabela de configuração

Nome do Campo	Tipo de Dados
login	TEXT
senha	TEXT

Fonte: elaboração do autor.

Como pode ser visto, é uma tabela simples que armazena as informações de login, sendo estas utilizadas no para o recebimento das informações atualizadas de dentro do sistema.

8.1.2 Tabela de Material de Apoio

A tabela de Material de apoio é extraida a partir do arquivo JSON de materiais transmitido pelo servidor, e é responsável por armazenar a lista de materiais para todas as disciplinas cursadas pelo acadêmico que está utilizando o aplicativo. Os campos e tipos de dados podem ser consultados na Tabela 14.

TABELA 14 - Layout da tabela de Material de Apoio

Nome do Campo	Tipo de Dados
nomeDisciplina	TEXT
publicacao	TEXT
nome	TEXT
descricao	TEXT
url	TEXT

Fonte: elaboração do autor.

Devido a quantidade de colunas ser reduzida, as informações dos materiais de apoio foi a única que não foi dividida em duas tabelas, sendo possível manter sem problemas as informações em apenas uma tabela, não gerando grande duplicidade nas informações.

8.1.3 Tabela de Horários do Semestre

Com o objetivo de facilitar as consultas SQL executadas na base de dados e fornecer uma complexidade menor na leitura do código, o JSON de Horários do Semestre retornado pelo servidor é extraido em duas tabelas unidas por uma chave. Desta forma, a tabela de Horários do Semestre é responsável pelo armazenamento das disciplinas e seus detalhes. Na Tabela 15 pode-se observar a lista dos campos e seus respectivos tipos.

TABELA 15 - Layout da tabela de Horários do Semestre

Nome do Campo	**Tipo de Dados**
codigo	TEXT
turma	TEXT
nome	TEXT
curso	TEXT
dataG2	TEXT
dataG3	TEXT
professor	TEXT
creditos	INTEGER
turno	TEXT
grade	INTEGER
periodo	INTEGER

Fonte: elaboração do autor.

8.1.4 Tabela de Horários do Semestre por Disciplina

Responsável por armazenar as informações dos horários das aulas e se as mesmas já ocorreram, esta tabela pode ser considerada uma extenção da tabela de Horários do Semestre, sendo ligada a mesma pelos campos código e turma. Na Tabela 16 são representados todos os campos da tabela e seus respectivos dados.

TABELA 16 - Layout da tabela de Horários do Semestre por Disciplina

Nome do Campo	Tipo de Dados
codigo	TEXT
turma	TEXT
hora	TEXT
diasemana	TEXT
data	TEXT
ocorreu	TEXT

Fonte: elaboração do autor.

8.1.5 Tabela de Notas da Graduação

Com o objetivo de facilitar as consultas SQL executadas na base de dados e fornecer uma complexidade menor na leitura do código, o JSON de Notas da Graduação retornado pelo servidor é extraido em duas tabelas unidas por uma chave. Desta forma, a tabela de Notas da Graduação é responsável pelo armazenamento das disciplinas e seus detalhes. Na Tabela 17 pode-se observar a lista dos campos e seus respectivos tipos.

TABELA 17 - Layout da tabela de Notas da Graduação

Nome do Campo	Tipo de Dados
nome	TEXT
notaG1	FLOAT
notaG3	FLOAT
notaG2	FLOAT
mediaFinal	FLOAT
estadoMateria	TEXT
statusAcademico	TEXT

Fonte: elaboração do autor.

8.1.6 Tabela de Avaliações

Responsável por armazenar as informações das avaliações já minstradas, esta tabela pode ser considerada uma extenção da tabela de Notas da Graduação, sendo ligada a mesma pelo campo nome. Na Tabela 18 são representados todos os campos da tabela e seus respectivos dados.

TABELA 18 - Layout da tabela de Avaliações

Nome do Campo	Tipo de Dados
nomeDisciplina	TEXT
peso	TEXT
nota	FLOAT
data	TEXT
nome	TEXT

Fonte: elaboração do autor.

8.2 Conexão ao Servidor e Extração dos Dados

A conexão com o servidor ocorre conforme explicado no item 5.4 deste trabalho, onde a partir de uma URL utilizando-se de um cliente HTTP para a conexão. Para o sincronismo das aplicações móveis, foi criado um servidor utilizando-se do endereço `http://minhaunomovel.no-ip.org`, sendo então necessário utilizar este endereço na montagem da URL de extração das informações. Os passos para conexão e extração dos dados estão representados no *Apêndice B.1*.

O processo de conexão com o servidor e de extração de dados ocorrem em conjunto onde a partir da conexão, caso a informação buscada seja retornada é chamada a função de extração de dados, que efetua a conversão do JSON transmitido pelo servidor em um objeto JavaScript, e posteriormente extrai as informações e insere as mesmas na base de dados.

Após o recebimento do arquivo JSON referente ao módulo do sistema a ser extraido, o arquivo JSON é convertido para um objeto nativo JavaScript, sendo que sua manipulação se da de forma extremamente simples, onde os vetores anteriormente representados no JSON agora são acessados como vetores da linguagem, e os objetos se tornam objetos realmente, sem necessidade de funções especiais para acessar seus dados. Desta forma, conforme pode ser visto no *Apêndice B.1* as informações preenchidas no objeto JSON são acessadas sem utilização de funções especiais de acesso e inseridas diretamente na base de dados SQLite.

8.3 Login e Persistência das Informações

Sendo o formulário de entrada do sistema, o formulário de login é a “porta de entrada” do acadêmico, sendo por meio deste feito a validação dos dados de login, e posteriormente o recebimento das informações. Na Figura 13 pode ser vista a interface deste formulário.

FIGURA 13 - Interface do Formulário de Login

Fonte: Do Autor.

Com o objetivo de ser um formulário simples, foi pensado em uma interface limpa, apenas com as informações necessárias. Desta forma o layout a cima foi o escolhido, apenas com os campos de usuário e senha, a logo da instituição e um botão de login.

Após inseridas as informações, no momento que é acionado o botão de login, a aplicação conecta-se com o servidor e recebe os dados do acadêmico. Também neste momento as informações de login e senha são armazenadas na tabela de configuração, sendo que exceto quando o usuário efetua logout (opção sair do menu principal da apliçação), o formulário de login não é mais exibido ao usuário, sendo feito o login de forma automática.

A implementação deste formulário pode ser consultada no *Apêndice B.2*.

8.4 Formulário Principal

Sendo o menu principal da aplicação o formulário principal é a tela de escolha do usuário, onde o mesmo pode acessar os outros menus da aplicação. Seguindo a idéia de simplicidade proposta, utiliza-se de uma lista de opções possuindo acima desta lista a logo da instituição, como pode ser visto na Figura 14.

A partir do formulário principal, o usuário pode ser levado para qualquer formulário

FIGURA 14 - Interface do Formulário Principal do Aplicativo.

Fonte: Do Autor.

da aplicação. A opção “Sobre” exibe um alerta mostrando mais informações sobre a apliçação, conforme pode ser visto na Figura 15.

FIGURA 15 - Informações sobre o Aplicativo.

Fonte: Do Autor.

A opção “Atualizar” exibida no menu principal é utilizada pelo usuário para receber novos dados do servidor, sendo assim atualizadas as informações armazenadas no dispositivo. Já a opção “Sair” é utilizada para fazer logout do sistema, excluindo as informações armazenadas na base de dados do aplicativo e retornando ao formulário de login.

A implementação deste formulário pode ser consultada no *Apêndice B.3*.

8.5 Material de Apoio

Desenvolvido com o objetivo de permitir a consulta da lista de materiais de apoio das disciplinas, e também permitir visualizar o material (desde que haja suporte pelo dispositivo para visualização do material disponibilizado). Na Figura 16 é possível visualizar o layout da interface gráfica do módulo.

FIGURA 16 - Interface do Formulário de Material de Apoio.

Fonte: Do Autor.

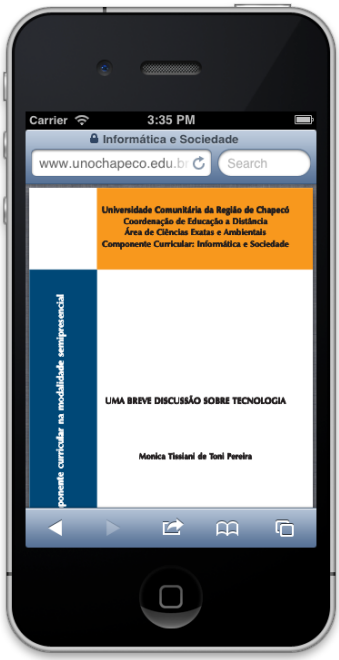
Ao ser selecionada alguma disciplina, a lista dos materiais da mesma é exibida. Para a Figura 17 foi selecionada a disciplina de Informática e Sociedade, carregando assim a lista de materiais da mesma.

FIGURA 17 - Interface do Formulário de Consulta de Materiais.

Fonte: Do Autor.

Ao se escolher uma disciplina, o navegador do dispositivo é chamado, para abrir o material escolhido pelo usuário. Neste ponto é solicitado o login do acadêmico no sistema acadêmico pelo browser, pois não possuímos acesso a sessão em que o arquivo foi extraido no servidor. Após o login, material é exibido, conforme Figura 18.

FIGURA 18 - Visualização do Material de Apoio selecionado pelo usuário.



Fonte: Do Autor.

A implementação dos formulários de exibição das disciplinas e dos materiais podem ser consultados nos Apêndices *B.4* e *B.5*.

8.6 Notas da Graduação

O formulário de Notas da Graduação é a interface do usuário para visualização, a partir das disciplinas cursadas pelo mesmo, visualizar suas médias e suas notas por avaliação. Sua interface incial, assim como os outros formulários de consulta, consiste na lista das disciplinas cursadas pelo acadêmico, conforme Figura 19.

FIGURA 19 - Visualização do Formulário de Notas da Graduação.

Fonte: Do Autor.

Ao selecionar uma disciplina, é apresentada uma tela com as médias do acadêmico. Caso a disciplina esteja aberta, a tela exibida é semelhante a visualizada na Figura 20. Já caso a disciplina ainda esteja aberta, a imagem exibida é semelhante a Figura 21, exibindo o botão detalhes para a consulta das avaliações feitas pelo acadêmico.

FIGURA 20 - Visualização das notas de uma disciplina fechada.

Fonte: Do Autor.

FIGURA 21 - Visualização das notas de uma disciplina em aberto.

Fonte: Do Autor.

Em caso de disciplina em aberto, onde o botão “Detalhes” é exibido, ao ser escolhio pelo usuário é exibida a tela de avaliações, que mostra as informações referentes as avaliações, caso as mesmas existam (Figura 22), ou é exibida uma mensagem informando que não existem avaliações (Figura 23).

FIGURA 22 - Visualização das atividades e suas informações.

Fonte: Do Autor.

FIGURA 23 - Alerta sobre não existirem avaliações cadastradas no sistema acadêmico.

Fonte: Do Autor.

A implementação dos formulários de exibição das disciplinas e das avaliações podem ser consultados nos Apêndices *B.6* e *B.7*.

8.7 Horários do Semestre

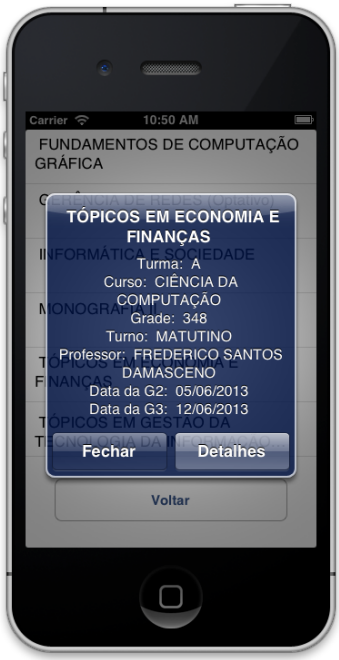
O formulário de horários do semestre exibe tanto informações detalhadas sobre a disciplina cursada pelo acadêmico como também os dias de aula da disciplina. Ao ser acessado, assim como os outros formulários, é exibida a lista das disciplinas cursadas, conforme mostrado na Figura 24.

FIGURA 24 - Visualização do Formulário de Horários do Semestre.

Fonte: Do Autor.

Ao selecionar uma disciplina na lista apresentada, é exibido ao acadêmico um alerta com as informações da disciplina (Figura 25) e também um botão detalhes, que exibe a lista completa de datas e horários de aula da disciplina escolhida (Figura 26).

FIGURA 25 - Visualização dos detalhes da disciplina selecionada.



Fonte: Do Autor.

FIGURA 26 - Visualização das datas e horários de aula.

Fonte: Do Autor.

Conforme pode ser visto na Figura 26, o último registro é exibido em destaque. Isto significa que a aquela aula ainda não foi realizada, assim como ocorre no sistema acadêmico Minha Uno.

A implementação dos formulários de exibição das disciplinas e dos horários podem

ser consultados nos Apêndices *B.8* e *B.9*.

8.8 Execução no Google Android

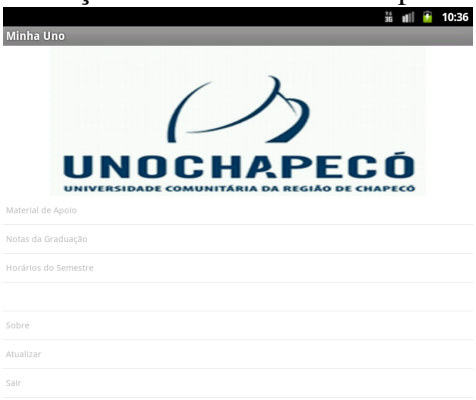
Conforme resultados obtidos no questionário, a maior parte dos acadêmicos que responderam o mesmo que possuem dispositivos móveis possuem o sistema operacional Android nos seus dispositivos. Apesar deste capítulo ter exibido, até o momento, apenas imagens do funcionamento da aplicação no sistema operacional iOS, a aplicação, sem nenhuma alteração em sua implementação, é executada também em dispositivos com o sistema operacional Android.

Conforme podem ser vistos nas Figuras 27, 28 e 29 as informações são exibidas apenas com um visual diferente, devido ao Titanium SDK gerar componentes gráficos nativos para a plataforma de destino.

FIGURA 27 - Visualização do Formulário de Login no Google Android.

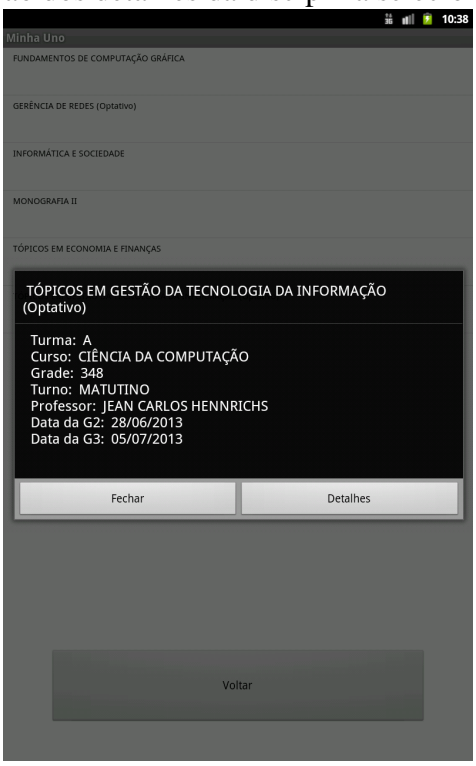
Fonte: Do Autor.

FIGURA 28 - Visualização do Formulário Principal no Google Android.



Fonte: Do Autor.

FIGURA 29 - Visualização dos detalhes da disciplina selecionada no Google Android.



Fonte: Do Autor.

8.9 Testes da aplicação

Devido a complexidade do desenvolvimento do servidor por não se possuir acesso direto aos dados dos acadêmicos da instituição, houve um atraso no desenvolvimento do projeto, o que não tornou possível o envio da aplicação para testes pelos acadêmicos que, no questionário, informaram que desejariam testar a aplicação. Para testes foram utilizados um dispositivo com iOS e dois dispositivos com Android, para assim verificar o comportamento da aplicação.

9 CONCLUSÃO

Este trabalho mostrou que é de interesse da comunidade acadêmica da instituição um aplicativo para dispositivos móveis, e que o desenvolvimento do mesmo também é possível. Este trabalho também mostra que mesmo não possuindo acesso direto as bases de dados computacionais de uma instituição, a partir de visões ou conexões a tabelas, é possível obter-se as informações do sistema acadêmico utilizando os meios onde as informações convencionalmente são disponibilizadas, neste caso, uma página da internet. Mesmo não sendo a melhor prática, com esforço e aplicação de conceitos é possível a obtenção dos dados desejados.

Apesar da falta de apoio da diretoria de T.I. da instituição para fornecimento das informações e infra-estrutura, com a utilização de métodos de extração e o esforço para entender o funcionamento da página web do sistema acadêmico foi possível extrair as informações utilizando algumas vulnerabilidades do sistema acadêmico atual para isso. Sem estas vulnerabilidades exploradas (como por exemplo o sistema de login utilizado) o trabalho tornaria-se mais difícil, e quem sabe não fosse possível obter as informações do sistema acadêmico Minha Uno.

Além disso, apesar das diversas tentativas de envio do questionário a toda a instituição, para assim obter-se dados conclusivos sobre todos os módulos do sistema acadêmico, apenas os estudantes de graduação da Área de Ciências Exatas e Ambientais receberam o questionário, desta forma não sendo possível atingir toda a instituição, principalmente o corpo de docentes, que posteriormente ao questionário ficaram sabendo da existência do mesmo.

Porém apesar de todas as dificuldades, o trabalho mostrou tanto a possibilidade de extrair-se as informações e implementar uma aplicação móvel, como também serviu de fonte de conhecimento para trabalhos similares, onde é possível, consultando-se este trabalho, serem desenvolvidos outros meios de integração ou novas aplicações para o sistema acadêmico Minha Uno, partindo-se da experiência obtida com esta aplicação para dispositivos móveis.

9.1 Trabalhos Futuros

Devido ao grau de complexidade do projeto não foi possível implementar todos os módulos do sistema acadêmico Minha Uno. Desta forma seria interessante a continuidade do projeto, implementando os módulos faltantes do perfil de graduação, e os outros perfis faltantes.

Para seguir as tendências de design atuais, seria muito interessante a adoção do design ”flat”na aplicação, implementando desta forma uma interface gráfica mais atrativa aos usuários.

Como este trabalho depende do layout do arquivo HTML do sistema acadêmico atual, caso o sistema acadêmico sofra alterações de layout é necessária manutenção no servidor, adequando a extração dos dados ao novo layout do arquivo HTML.

Com o objetivo de a aplicação tornar-se opensource e poder atingir outras universidades, seria interessante padronizar a integração da aplicação e melhorar a sua abstração, possibilitando assim as universidades a partir do projeto base, sem muitas alterações no mesmo, poderem utilizar a aplicação para benefício dos seus acadêmicos.

Implementar notificações automáticas na aplicação, para avisar os usuários quando ocorre alguma alteração nos dados do mesmo, como o recebimento de um novo material ou uma nova nota cadastrada no sistema acadêmico.

Como dados são transmitidos entre o servidor e a aplicação, é necessário implementar medidas de segurança, para que os dados não possam ser interceptados no momento do sincronismo.

REFERÊNCIAS

APPCELERATOR. **Titanium Cross Platform Mobile App SDK**. 2013. Disponível em: <http://www.appcelerator.com/platform/titanium-sdk>. Acesso em: 09 jun. 2013.

APPLE. **iOS**: O sistema operacional móvel mais avançado do mundo. 2013. Disponível em: <http://www.apple.com/br/ios/what-is/>. Acesso em: 9 jun. 2013.

CARVALHO, J. et al. **Sistemas Operacionais para Dispositivos Móveis**. 2010. Disponível em: <http://goo.gl/uQxqa>. Acesso em: 30 nov. 2012.

COSTA, N. P. de O.; FILHO, N. F. D.; DUARTE, A. F. **Avaliação comparativa de sistemas operacionais para dispositivos móveis**: Foco em suas funcionalidades. 2012. Disponível em: <http://www.tecsi.fea.usp.br/envio/contecsi/index.php/envio/rt/metadata/196/0>. Acesso em: 04 out. 2012.

CROCKFORD, D. **JavaScript**: The world's most misunderstood programming language. 2001. Disponível em: <http://javascript.crockford.com/javascript.html>. Acesso em: 8 jun. 2013.

CROCKFORD, D. **Introdução ao JSON**. 2013. Disponível em: <http://www.json.org/json-pt.html>. Acesso em: 8 jun. 2013.

DEITEL, P.; DEITEL, H. **Java**: Como programar 8ªedição. São Paulo: Pearson, 2010.

FACEBOOK. **Desenvolvedores do Facebook**. 2012. Disponível em: <https://developers.facebook.com/docs/>. Acesso em: 15 nov. 2012.

FIELDING, R. T. **Architectural Styles and the Design of Network-based Software Architectures**. 2000. Disponível em: <http://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation_2up.pdf>. Acesso em: 5 jun. 2013.

FLURRY. **iOS and Android Adoption Explodes Internationally**. ago 2012. Disponível em: <http://blog.flurry.com/bid/88867/iOS-and-Android-Adoption-Explodes-Internationally>. Acesso em: 15 nov. 2012.

FUNDESTE. **Balanço Social 2011**. 2011. Disponível em: <http://www.fundeste.org.br/index.php/balanco_social_2011>. Acesso em: 15 nov. 2012.

GOOGLE. **Android 4.2: A new flavor of Jelly Bean**. 2013.

HARTMANN, G.; STEAD, G.; DEGANI, A. **Cross-platform mobile development**. 2011. Disponível em: <http://www.mole-project.net/images/documents/

deliverables/WP4_crossplatform_mobile_development_March2011.pdf>. Acesso em: 15 nov. 2012.

HEDLEY, J. **jsou Java HTML Parser**. 2013. Disponível em: <http://www.jsoup.org>. Acesso em: 8 jun. 2013.

HUAWEI. **Balanço Huawei da Banda Larga 1T2012**. 2012. Disponível em: <http://www.huawei.com/ilink/br/download/HW_193137>. Acesso em: 15 nov. 2012.

IEEE. **Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications**. 2012. Disponível em: <http://standards.ieee.org/getieee802/download/802.11-2012.pdf>. Acesso em: 15 nov. 2012.

JUNIOR, J. B. B.; COUTINHO, C. P. **The use of Mobile Technologies by the Portuguese Academic Community**: An exploratory survey. 2008. Disponível em: <http://repositorium.sdum.uminho.pt/bitstream/1822/7817/1/Iadis%25202008.pdf>. Acesso em: 15 nov. 2012.

LECHETA, R. R. **Google ANDROID**: Aprenda a criar aplicações para dispositivos móvies com android sdk. São Paulo: NOVATEC, 2009. ISBN 9788575222447.

LIISBERG, N. **icebreakrest**. 2010. Disponível em: <https://code.google.com/p/icebreakrest/>. Acesso em: 8 jun. 2013.

MAYANI, P. **Android – JSON Parsing example**. 2011. Disponível em: <http://www.technotalkative.com/android-json-parsing/>. Acesso em: 8 jun. 2013.

NOKIA. **Nokia Corporation Q4 and full year 2012 Interim Report**. 2013. Disponível em: <http://www.results.nokia.com/results/Nokia_results2012Q4e.pdf>.

OHA. **Alliance Members**. 2012. Disponível em: <http://www.openhandsetalliance.com/oha_members.html>. Acesso em: 30 nov. 2012.

RAMOS, M. F. **A evolução da telefonia celular**. 2012. Disponível em: <http://www.slideshare.net/slideshow/embed_code/10858413>.

REGO, P. **A Cronologia das Gerações de Telefonia Móvel. 1G 2G 3G e 4G**. 2011. Disponível em: <http://www.mobilepronto.org/_blog/Blog/post/A_Cronologia_das_Gera%C3%A7%C3%B5es_de_Telefonia_M%C3%B3vel_1G_2G_3G_e_4G/>. Acesso em: 15 nov. 2012.

SHKLAR, L.; ROSEN, R. **Web Application Architecture**: Principles, protocols and practices. New Jersey: John Wiley & Sons Ltd, 2003. 314-322 p.

SQLITE. **About SQLite**. 2013. Disponível em: <http://www.sqlite.org/, urlaccessdate = 8 jun. 2013,>.

TANENBAUM, A. S. **Sistemas Operacionais Modernos**. Rio de Janeiro: Prentice Hall Brasil, 2010. ISBN 9788576052371.

UFLA. **O que é uma rede sem fio?** 2009. Disponível em: <http://semfio.ufla.br/index.php?option=com_content&view=article&id=63>. Acesso em: 9 jun. 2013.

W3C. **World Wide Web Consortium**. 2012. Disponível em: <http://www.w3.org/>. Acesso em: 15 nov. 2012.

WEISSTEIN, E. W. **Student's t-Distribution**. 2012. Disponível em: <http://mathworld.wolfram.com/Studentst-Distribution.html>. Acesso em: 30 nov. 2012.

APÊNDICE A – Servidor

Abaixo serão apresentados os códigos desenvolvidos para o servidor que oferece suporte a aplicação.

A.1 Login.java

```
package edu.unochapeco.saa;

import java.io.IOException;
import org.jsoup.Connection;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;

public class Login {

    private String session = null;


    public boolean connect(String user, String pass) throws IOException
    {
        String url = "https://www.unochapeco.edu.br/usuarios/login";

        if (this.session == null) {
            Connection.Response response = Jsoup.connect(url)
                    .data("login_submited", "1",
                    "usuario", user,
                    "senha", pass,
                    "submit", "entrar")
                    .method(Connection.Method.POST)
                    .execute();

            this.session = response.cookie("PHPSESSID");
        }

         //Testa se o login foi bem sucedido
```

```
            Document document = Jsoup.connect(url)
                .cookie("PHPSESSID", this.session)
                .get();

        return !document.text().contains("GRADUAÇÃO PÓS BOLSAS NOTÍCIAS UNOWEBTV "
                + "EVENTOS INSTITUCIONAL MINHA UNO WEBMAIL CONTATO");


    }

    public String getSession() throws IOException {

        return this.session;
    }
}
```

A.2 MaterialApoio.java

```
package edu.unochapeco.saa;

import java.io.IOException;
import org.json.JSONArray;
import org.json.JSONObject;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.nodes.Element;
import org.jsoup.select.Elements;

public class MaterialApoio {

    private String session = null;

    public MaterialApoio(String session) {
        this.session = session;
    }

    public String getMateriais() throws IOException {
        String url = "https://www.unochapeco.edu.br/saa/materialApoio.php";

        Document document = Jsoup.connect(url)
                .cookie("PHPSESSID", session)
                .timeout(8000)
                .get();

        Elements disciplinas = document.select("form tr td:eq(1) a");
```

```
        JSONArray disciplinaArray = new JSONArray();
        for (Element disciplina : disciplinas) {
            JSONObject disciplinaObject = new JSONObject();
            disciplinaObject.put("nome", disciplina.text());
            disciplinaObject.put("materiais", parse(disciplina.attr("href")));

            disciplinaArray.put(disciplinaObject);
        }
        return new JSONObject().put("disciplinas", disciplinaArray).toString();
    }

    private JSONArray parse(String url) throws IOException {
        String base = "https://www.unochapeco.edu.br";

        Document document = Jsoup.connect(base + url)
                .cookie("PHPSESSID", session)
                .timeout(8000)
                .get();

        Elements arquivos = document.select("form tr:contains(Arquivo) a");
        Elements publicacoes = document
                .select("form tr:contains(Publicação) td:eq(1)");
        Elements descricoes = document
                .select("form tr:contains(Descrição) td:eq(1)");

        JSONArray materialArray = new JSONArray();
        for (int i = 0; i < publicacoes.size(); i++) {
            JSONObject materialObject = new JSONObject();

            materialObject.put("nome", arquivos.get(i + 1).text());
            materialObject.put("url", base + arquivos.get(i + 1).attr("href"));
            materialObject.put("publicacao", publicacoes.get(i).text());
            materialObject.put("descricao", descricoes.get(i).text());

            materialArray.put(materialObject);
        }

        /*
         * Link para fazer o download de todos os arquivos (.zip)
         */
        try {
            Element todos = document.select("form tr a")
                    .select(":contains(.zip)").last();

            materialArray.put(new JSONObject()
                    .put("nome", "Todos os arquivos")
                    .put("url", base + todos.attr("href"))
                    .put("publicacao", "")
                    .put("descricao", todos.text()));
        } catch (NullPointerException e) {
```

```
            ;
        }
        return materialArray;
    }
}
```

A.3 NotasGraduacao.java

```
package edu.unochapeco.saa;

import java.io.IOException;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
import org.json.JSONArray;
import org.json.JSONObject;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.select.Elements;
import org.jsoup.nodes.Element;

public class NotasGraduacao {

    private String session = null;

    public NotasGraduacao(String session) {
        this.session = session;
    }

    public String getNotas() throws IOException {
        String url = "https://www.unochapeco.edu.br/saa/notas.php";

        Document document = Jsoup.connect(url)
                .cookie("PHPSESSID", session)
                .timeout(8000)
                .get();

        Elements disciplinas = document.select("form table:eq(0) tr td:eq(1) a");

        JSONArray disciplinaArray = new JSONArray();
        for (Element disciplina : disciplinas) {
            JSONObject disciplinaObject = new JSONObject();
            disciplinaObject.put("nome", disciplina.text());
            disciplinaObject.put("dados", parse(disciplina.attr("href")));

            disciplinaArray.put(disciplinaObject);
        }
```

```
        return new JSONObject().put("disciplinas", disciplinaArray).toString();
    }

    public JSONObject parse(String url) throws IOException {
        String base = "https://www.unochapeco.edu.br/saa/";

        Document document = Jsoup.connect(base + url)
                .cookie("PHPSESSID", session)
                .timeout(8000)
                .get();

        if (!document.select(":contains(Disciplina Fechada!)").isEmpty()) {

            JSONObject avaliacaoObject = new JSONObject();
            document = Jsoup.connect(base + "notas.php")
                    .cookie("PHPSESSID", session)
                    .timeout(8000)
                    .get();

            Pattern pattern = Pattern.compile("coddisc=[0-9]+&");
            Matcher matcher = pattern.matcher(url);

            String coddisc = null;
            while (matcher.find()) {
                coddisc = matcher.group().replaceAll("\\D", "");
                break;
            }

            Element elemento = document.select("form[name\$=graduacao] tr"
                    + ":contains(" + coddisc + ")").first();

            avaliacaoObject.put("estado", "fechada");
            avaliacaoObject.put("status", elemento.select("td:eq(11)").text());
            avaliacaoObject.put("G1", elemento.select("td:eq(6)").text());
            avaliacaoObject.put("G2", elemento.select("td:eq(7)").text());
            avaliacaoObject.put("G3", elemento.select("td:eq(8)").text());
            avaliacaoObject.put("MF", elemento.select("td:eq(9)").text());

            return avaliacaoObject;
        }

        Elements avaliacoes = document.select("form table:eq(0) tr:gt(4)");

        JSONArray avaliacaoArray = new JSONArray();
        for (Element avaliacao : avaliacoes) {
            if (!avaliacao.select("td:eq(3):contains(nota)").isEmpty()
                    || avaliacao.select("td:eq(3)").isEmpty()) {
                continue;
            }
```

```
            JSONObject avaliacaoObject = new JSONObject();
            avaliacaoObject.put("nome", avaliacao.select("td:eq(0)").text());
            avaliacaoObject.put("peso", avaliacao.select("td:eq(1)").text());
            avaliacaoObject.put("data", avaliacao.select("td:eq(2)").text());
            avaliacaoObject.put("nota", avaliacao.select("td:eq(3)").text());

            avaliacaoArray.put(avaliacaoObject);
        }

        String mediaG1 = new String();
        String mediaG2 = new String();
        try {
            mediaG1 = document
                    .select("form table:eq(0) tr:contains(Média de G1) td:eq(1)")
                    .first()
                    .text();

            mediaG2 = document
                    .select("form table:eq(0) tr:contains(Média de G2) td:eq(1)")
                    .first()
                    .text();
        } catch (NullPointerException e) {
            ;
        }

        return new JSONObject()
                .put("estado", "aberta")
                .put("mediaG1", mediaG1)
                .put("mediaG2", mediaG2)
                .put("avaliacoes", avaliacaoArray);
    }
}
```

A.4 HorariosSemestre.java

```
package edu.unochapeco.saa;

import java.io.IOException;
import org.json.JSONArray;
import org.json.JSONObject;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.nodes.Element;
import org.jsoup.select.Elements;

public class HorariosSemestre {
```

```
private String session = null;

public HorariosSemestre(String session) {
    this.session = session;
}

public String getHorarios() throws IOException {
    String url = "https://www.unochapeco.edu.br/saa/hor_aluno.php";

    Document document = Jsoup.connect(url)
            .cookie("PHPSESSID", session)
            .timeout(8000)
            .get();

    Elements codigos = document.select("form tr:gt(1) td:eq(0) a");
    Elements nomes = document.select("form tr:gt(1) td:eq(1) a");
    Elements turmas = document.select("form tr:gt(1) td:eq(2) a");

    JSONArray disciplinasArray = new JSONArray();
    for (int i = 0; i < codigos.size(); i++) {
        JSONObject disciplinaObject = new JSONObject();
        disciplinaObject.put("codigo", codigos.get(i).text());
        disciplinaObject.put("nome", nomes.get(i).text());
        disciplinaObject.put("turma", turmas.get(i).text());
        disciplinaObject.put("dados", parse(nomes.get(i).attr("href")));

        disciplinasArray.put(disciplinaObject);
    }
    return new JSONObject().put("disciplinas", disciplinasArray).toString();
}

public JSONObject parse(String url) throws IOException {
    String base = "https://www.unochapeco.edu.br/saa/";

    Document document = Jsoup.connect(base + url)
            .cookie("PHPSESSID", session)
            .timeout(8000)
            .get();

    String[] detalhes = {"curso", "grade", "disciplina", "período",
        "professor", "turno", "créditos", "g2", "g3"};

    JSONObject detalhesObject = new JSONObject();
    try {
        for (String detalhe : detalhes) {
            Element element = document
                    .select("form tr[bgcolor]:contains(" + detalhe
                    + ") td:eq(1)").first();
```

```
                detalhesObject.put(detalhe, element.text());
            }
        } catch (NullPointerException e) {
            ;
        }

        JSONArray horariosArray = new JSONArray();
        try {
            Elements horarios = document.select("form table:eq(1) tr:gt(1)");
            for (Element horario : horarios) {
                JSONObject horarioObject = new JSONObject();

                horarioObject.put("dia", horario.select("td:eq(0)").text());
                horarioObject.put("semana", horario.select("td:eq(1)").text());
                horarioObject.put("hora", horario.select("td:eq(2)").text());

                if (!horario.select("b").isEmpty()) {
                    horarioObject.put("ocorreu", "false");
                } else {
                    horarioObject.put("ocorreu", "true");
                }

                horariosArray.put(horarioObject);
            }
        } catch (NullPointerException e) {
            ;
        }
        return new JSONObject()
                .put("detalhes", detalhesObject)
                .put("horarios", horariosArray);
    }
}
```

A.5 Main.java

```
package edu.unochapeco.saa;

import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;

public class Main {

    public static void main(String[] args) {
        RestServer rest; // Declare the HTTP server class
```

```
try {
    rest = new RestServer();
    rest.setPort(90);

    while (true) {
        rest.getHttpRequest();
        String usuario = rest.getQuery("usuario");
        String senha = rest.getQuery("senha");
        String info = rest.getQuery("info");


        if(usuario != null || senha != null)
        {
            Login login = new Login();
            boolean loginValido = login.connect(usuario, senha);
            String sessao = login.getSession();
            String JSONnotas, JSONmaterial, JSONhorarios;
            NotasGraduacao notas = new NotasGraduacao(sessao);
            MaterialApoio material = new MaterialApoio(sessao);
            HorariosSemestre horarios = new HorariosSemestre(sessao);

            if(!loginValido)
            {
                System.out.println("Login Invalido");
                rest.write("0");
                continue;
            }

            else if(info == null)
            {
                rest.write("1");
                continue;
            }
            else if(info.equalsIgnoreCase("notas"))
            {
               System.out.println(usuario + ": Notas");
               JSONnotas = notas.getNotas();
               rest.write(JSONnotas + "\n");
            }
            else if (info.equalsIgnoreCase("materiais"))
            {
                System.out.println(usuario + ": Materiais");
                JSONmaterial = material.getMateriais();
                rest.write(JSONmaterial + "\n");
            }
            else if(info.equalsIgnoreCase("horarios"))
            {
                System.out.println(usuario + ": Horarios");
                JSONhorarios = horarios.getHorarios();
                rest.write(JSONhorarios + "\n");
```

```
                    }
                    else
                    {
                        System.out.println("Opcao invalida");
                        rest.write("Opção Inválida");
                    }

                    Thread.sleep(1);
                }
                else {
                    rest.write("Parâmetros Inválidos");
                }
            }
        }
        catch (IOException ex) {
            System.out.println(ex.getMessage());
        } catch (InterruptedException ex) {
            Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

APÊNDICE B – Aplicativo

Abaixo serão apresentados os códigos desenvolvidos para a aplicação, onde é exibido o consumo dos dados e a criação das interfaces gráficas e suas funcionalidades.

B.1 conexao.js

```
var ConexaoServidor = function(_Usuario, _Senha)
{
	var urlBase = "http://minhaunomovel.no−ip.org:90?usuario=" + _Usuario + "&senha=" + _Senha;
	var database, databaseName = 'MinhaUnoDB';

	var validarLogin = function(_EventoLoginValido)
	{
		var client = Ti.Network.createHTTPClient({
	// function called when the response data is available
	onload : function(e) {
		if(this.responseText == '1')
		    _EventoLoginValido();
	},
	onerror : function(e) {
		msgBox('Atenção', 'Ocorreu um problema na comunicação com o servidor. Verifique sua conexão com a internet ou tente
		    novamente mais tarde.');
		Ti.API.debug(e.error);
	},
	timeout : 50000
		});

		client.open("GET", urlBase);
		client.send();
	}

	var inicializarDatabase = function()
	{
		database = Ti.Database.open(databaseName);
```

```
        //Criação das tabelas feitas desta forma pois não funcionou a passagem de variáveis com o SQL para a função execute.
        database.execute('CREATE TABLE IF NOT EXISTS HorarioSemestre(codigo TEXT, turma TEXT, nome TEXT, curso
            TEXT, dataG2 TEXT, dataG3 TEXT, professor TEXT, creditos INTEGER, turno TEXT, grade INTEGER, periodo
            INTEGER);');
        database.execute('CREATE TABLE IF NOT EXISTS HorarioSemestreDisciplina(codigo TEXT, turma TEXT, hora TEXT,
             diasemana TEXT, data TEXT, ocorreu TEXT);');
        database.execute('CREATE TABLE IF NOT EXISTS NotaGraduacao(nome TEXT, notaG3 FLOAT, mediaFinal FLOAT,
            notaG2 FLOAT, notaG1 FLOAT, estadoMateria TEXT, statusAcademico TEXT);');
        database.execute('CREATE TABLE IF NOT EXISTS NotaGraduacaoAvaliacao(nomeDisciplina TEXT, peso TEXT, nota
            FLOAT, data TEXT, nome TEXT);');
        database.execute('CREATE TABLE IF NOT EXISTS MaterialApoio(nomeDisciplina TEXT, publicacao TEXT, nome
            TEXT, descricao TEXT, url TEXT);');

        database.close();
}

var extrairMaterialApoio = function(_url, _proximaFuncao)
{
        var client = Ti.Network.createHTTPClient({
onload : function(e) {
        var json = JSON.parse(this.responseText);
                var i, j, disciplina, detalhes;

                database = Ti.Database.open(databaseName);
                database.execute('delete from MaterialApoio;');

                for (i = 0; i < json.disciplinas.length; i++) {
                        disciplina = json.disciplinas[i];

                        for(j = 0; j < disciplina.materiais.length; j++){
                                detalhe = disciplina.materiais[j];
                                database.execute('insert into MaterialApoio (nomeDisciplina, publicacao, nome, descricao, url)
                                        values (?,?,?,?,?);', disciplina.nome, detalhe.publicacao, detalhe.nome, detalhe.descricao,
                                        detalhe.url);
                        }
                }

                database.close();

        if(_proximaFuncao)
                _proximaFuncao();
},
onerror : function(e) {
        msgBox('Atenção', 'Ocorreu um problema na comunicação com o servidor. Verifique sua conexão com a internet ou tente
            novamente mais tarde.');
        Ti.API.debug(e.error);
},
timeout : 50000
        });
```

```
        client.open("GET", _url);
        client.send();
}

var extrairNotasGraduacao = function(_url, _proximaFuncao)
{
        var client = Ti.Network.createHTTPClient({
onload : function(e) {
        var json = JSON.parse(this.responseText);
                var i, j, disciplina, detalhes;

                database = Ti.Database.open(databaseName);
                database.execute('delete from NotaGraduacao;');
                database.execute('delete from NotaGraduacaoAvaliacao');

                for (i = 0; i < json.disciplinas.length; i++) {
                        disciplina = json.disciplinas[i];

                        if(disciplina.dados.estado == 'fechada')
                        {
                                database.execute('insert into NotaGraduacao (nome, notaG3, mediaFinal, notaG2, notaG1,
                                        estadoMateria, statusAcademico) values (?,?,?,?,?,?,?);', disciplina.nome, disciplina.dados.
                                        G3, disciplina.dados.MF, disciplina.dados.G2, disciplina.dados.G1, disciplina.dados.
                                        estado, disciplina.dados.status);
                        }
                        else
                        {
                                database.execute('insert into NotaGraduacao (nome, notaG3, mediaFinal, notaG2, notaG1,
                                        estadoMateria, statusAcademico) values (?,?,?,?,?,?,?);', disciplina.nome, 0, 0, disciplina.
                                        dados.mediaG2, disciplina.dados.mediaG1, disciplina.dados.estado, "");

                                for (j = 0; j < disciplina.dados.avaliacoes.length; j++)
                                {
                                        database.execute('insert into NotaGraduacaoAvaliacao (nomeDisciplina, peso, nota,
                                                data, nome) values (?,?,?,?,?);', disciplina.nome, disciplina.dados.avaliacoes[j].
                                                peso, disciplina.dados.avaliacoes[j].nota, disciplina.dados.avaliacoes[j].data,
                                                disciplina.dados.avaliacoes[j].nome);
                                }
                        }

                }

                database.close();

        if(_proximaFuncao)
                _proximaFuncao();
},
onerror : function(e) {
        msgBox('Atenção', 'Ocorreu um problema na comunicação com o servidor. Verifique sua conexão com a internet ou tente
                novamente mais tarde.');
```

```
        Ti.API.debug(e.error);
},
timeout : 50000
        });

        client.open("GET", _url);
        client.send();
}

var extrairHorariosSemestre = function(_url, _proximaFuncao)
{
        var client = Ti.Network.createHTTPClient({
// function called when the response data is available
onload : function(e) {
                var json = JSON.parse(this.responseText);
        var i, j, disciplina, detalhes;

                database = Ti.Database.open(databaseName);
                database.execute('delete from HorarioSemestre;');
                database.execute('delete from HorarioSemestreDisciplina');

                for (i = 0; i < json.disciplinas.length; i++) {
                        disciplina = json.disciplinas[i];


                        database.execute('insert into HorarioSemestre(codigo, turma, nome, curso, dataG2, dataG3, professor,
                                creditos, turno, grade, periodo) values (?,?,?,?,?,?,?,?,?,?,?);', disciplina.codigo, disciplina.turma,
                                disciplina.nome, disciplina.dados.detalhes.curso, disciplina.dados.detalhes.g2, disciplina.dados.
                                detalhes.g3, disciplina.dados.detalhes.professor, disciplina.dados.detalhes.creditos, disciplina.dados
                                .detalhes.turno, disciplina.dados.detalhes.grade, disciplina.dados.detalhes.periodo);

                        for(j = 0; j < disciplina.dados.horarios.length; j++)
                        {
                                var horario = disciplina.dados.horarios[j];

                                database.execute('insert into HorarioSemestreDisciplina(codigo, turma, hora, diasemana, data,
                                        ocorreu) values (?,?,?,?,?,?);', disciplina.codigo, disciplina.turma, horario.hora, horario.
                                        semana, horario.dia, horario.ocorreu);
                        }

                }

                database.close();

        if(_proximaFuncao)
                _proximaFuncao();
},
onerror : function(e) {
        msgBox('Atenção', 'Ocorreu um problema na comunicação com o servidor. Verifique sua conexão com a internet ou tente
                novamente mais tarde.');
```

```
                Ti.API.debug(e.error);
        },
        timeout : 50000
                });

                client.open("GET", _url);
                client.send();
        }

        this.extrairInformacoes = function(_EventoFimExtracao)
        {
                var urlMaterialApoio = urlBase + '&info=materiais';
                var urlNotasGraduacao = urlBase + '&info=notas';
                var urlHorariosSemestre = urlBase + '&info=horarios';

                inicializarDatabase();

                validarLogin(function(){
                        extrairMaterialApoio(urlMaterialApoio, extrairHorariosSemestre(urlHorariosSemestre, extrairNotasGraduacao(
                                urlNotasGraduacao, _EventoFimExtracao)));
                });
        }
};
```

B.2 Login.js

```
Ti.include('Funcoes.js');
Ti.include('conexao.js');

function FormLogin()
{
        //Variáveis de controle dos tamanhos
        var LarguraEdit = calcularProporcaoLarguraTela(80);
        var AlturaEdit = calcularProporcaoAlturaTela(10);
        var LarguraLabel = calcularProporcaoLarguraTela(80);
        var AlturaLabel = calcularProporcaoAlturaTela(8);
        var DistanciaBorda = calcularProporcaoLarguraTela(10);
        var AlturaImagem = calcularProporcaoAlturaTela(20);
        var LarguraImagem = calcularProporcaoLarguraTela(80);

        //Variáveis Diversas
        var form; //Formulário Base
        var edtLogin, edtSenha; //Edits de Usuário e Senha
        var lblLogin, lblSenha; //Label de Usuario e Senha
        var scroll;
        var imgUnochapeco;
```

```
var btnLogin;

this.create = function()
{
	var db = Ti.Database.open('MinhaUnoDB')
	db.execute('CREATE TABLE IF NOT EXISTS Configuracao(login TEXT, senha TEXT);');
	var configuracoes = db.execute('select login, senha from configuracao;');
	if(!configuracoes.rowCount)
	{
		form = Titanium.UI.createWindow({
		title:'Login',
		backgroundColor:'#fff'
		});

		scroll = Ti.UI.createScrollView({
			contentWidth:'auto',
			contentHeight:'auto',
			top: 0,
			showVerticalScrollIndicator: true
		});

		imgUnochapeco = Ti.UI.createImageView({
			image:'LogoUno.png',
		height: AlturaImagem,
		width: LarguraImagem,
		top: calcularProporcaoAlturaTela(5),
		left: DistanciaBorda
		});

		lblUsuario = Ti.UI.createLabel
		({
			text: 'Usuário',
		    textAlign:'center',
		    top: imgUnochapeco.top + AlturaImagem + calcularProporcaoAlturaTela(2),
			height: AlturaLabel,
		    width: LarguraLabel,
		    left: DistanciaBorda
		});

		edtUsuario = Ti.UI.createTextField
		({
			top: lblUsuario.top + AlturaLabel,
		    height: AlturaEdit,
		    width: LarguraEdit,
		    left: DistanciaBorda,
		    enableReturnKey:false,
			borderStyle:Titanium.UI.INPUT_BORDERSTYLE_ROUNDED,
			returnKeyType:Titanium.UI.RETURNKEY_DEFAULT
		});
```

```
lblSenha = Ti.UI.createLabel
({
        text: 'Senha',
    textAlign:'center',
    top: edtUsuario.top + AlturaEdit + calcularProporcaoAlturaTela(1),
    height: AlturaLabel,
    width: LarguraLabel,
    left: DistanciaBorda
});

edtSenha = Ti.UI.createTextField
({
        width: LarguraEdit,
        height: AlturaEdit,
        top: lblSenha.top + AlturaLabel,
    left: DistanciaBorda,
    passwordMask:true,
    enableReturnKey:false,
        borderStyle:Titanium.UI.INPUT_BORDERSTYLE_ROUNDED,
        returnKeyType:Titanium.UI.RETURNKEY_DEFAULT
});

//Botoes
btnLogin = Ti.UI.createButton
({
        width: LarguraEdit,
        height: AlturaEdit,
        top: edtSenha.top + AlturaEdit + calcularProporcaoAlturaTela(10),
    left: DistanciaBorda,
    title: 'Login'
});

btnLogin.addEventListener('click', function(e){
        var Conexao = new ConexaoServidor(edtUsuario.value, edtSenha.value);
        Conexao.extrairInformacoes(function(){
                db.execute('insert into configuracao(login, senha) values (?,?);', edtUsuario.value, edtSenha.
                        value);
                var win = Titanium.UI.createWindow({
                url: 'FormPrincipal.js',
                backgroundColor: 'white'
                });

                win.open();
                form.close();
        });
});

scroll.add(imgUnochapeco);
scroll.add(lblUsuario);
scroll.add(edtUsuario);
```

```
                scroll.add(lblSenha);
                scroll.add(edtSenha);
                scroll.add(btnLogin);
                form.add(scroll);
                form.open();
        }
        else
        {
                var win = Titanium.UI.createWindow({
                url: 'FormPrincipal.js',
                backgroundColor: 'white'
                });

                win.open();
        }
    };

}
```

B.3 FormPrincipal.js

```
Ti.include('Funcoes.js');
Ti.include('conexao.js');
Ti.include('FormLogin.js');

var Opcoes = [{title: 'Material de Apoio'},
              {title: 'Notas da Graduação'},
              {title: 'Horários do Semestre'},
              {title: ''},
              {title: 'Sobre'},
              {title: 'Atualizar'},
              {title: 'Sair'}
];

var AlturaImagem = calcularProporcaoAlturaTela(20);
var LarguraImagem;
var DistanciaBorda;
var AlturaGrid = calcularProporcaoAlturaTela(79)

if(Ti.Platform.model.indexOf('iPad') !== -1)
{
    LarguraImagem = calcularProporcaoLarguraTela(60);
    DistanciaBorda = calcularProporcaoLarguraTela(20);
}
else
{
```

```
        LarguraImagem = calcularProporcaoLarguraTela(80);
        DistanciaBorda = calcularProporcaoLarguraTela(10);
}

var imgUnochapeco = Ti.UI.createImageView
({
        image:'LogoUno.png',
    height: AlturaImagem,
    width: LarguraImagem,
    top: calcularProporcaoAlturaTela(0),
    left: DistanciaBorda
});

// create table view
var tableview = Titanium.UI.createTableView({
        top: calcularProporcaoAlturaTela(20),
        height: AlturaGrid,
    data:Opcoes,
});

// create table view event listener
tableview.addEventListener('click', function(e)
{
    // event data
    var index = e.index;

    /*if(index == 0)
        {
            Ti.include('FormConsultaMaterial.js');
        }
        else if(index == 1)
        {
            Ti.include('FormConsultaNota.js');
        }
        else if(index == 2)
        {
            Ti.include('FormConsultaHorarioSemestre.js');
        }
        else if(index == 4)
        {
            msgBox('Sobre', 'Trabalho de conclusão do curso de Ciência da Computação 2013/1. \n\nAcadêmico: Andrei Jiácomo
                Zuse \nOrientador: Marcelo Cezar Pinto');
        }*/
        switch(index)
        {
            case 0:
                Ti.include('FormConsultaMaterial.js');
                break;
            case 1:
                Ti.include('FormConsultaNota.js');
```

```
                    break;
                case 2:
                    Ti.include('FormConsultaHorarioSemestre.js');
                    break;
                case 4:
                    msgBox('Sobre', 'Trabalho de conclusão do curso de Ciência da Computação 2013/1. \n\nAcadêmico: Andrei Jiá
                        como Zuse \nOrientador: Marcelo Cezar Pinto');
                    break;
                case 5:
                    var db = Ti.Database.open('MinhaUnoDB')
                    var configuracoes = db.execute('select login, senha from configuracao;');
                    var usuario = configuracoes.fieldByName('login');
                    var senha = configuracoes.fieldByName('senha');
                    configuracoes.close();
                    db.close();
                    Ti.API.info(usuario);
                    Ti.API.info(senha);
                    var Conexao = new ConexaoServidor(usuario, senha);
                    Conexao.extrairInformacoes(alert('Terminou'));
                    break;
                case 6:
                    var db = Ti.Database.open('MinhaUnoDB')
                    var configuracoes = db.execute('delete from configuracao;');
                    db.close();
                    var login = new FormLogin();
                    login.create();
                    break;
        }

});

Ti.UI.currentWindow.add(imgUnochapeco);
Ti.UI.currentWindow.add(tableview)
```

B.4 FormConsultaMaterial.js

```
Ti.include('Funcoes.js');
Ti.include('FormConsultaMaterialDisciplina.js');

var dados = [];

var formConsultaMaterial = Titanium.UI.createWindow({
        top: 0,
        height: pegarAlturaTela(),
        width: pegarLarguraTela(),
        backgroundColor: 'white'
```

```
});

//Carrega as informações da tabela no vetor de dados para inserir no tableView
var db = Ti.Database.open('MinhaUnoDB')
var disciplinas = db.execute('SELECT distinct NomeDisciplina FROM MaterialApoio ORDER BY NomeDisciplina');
while (disciplinas.isValidRow())
{
	var row = Ti.UI.createTableViewRow({
		height: '80dp',
		id: disciplinas.fieldByName('NomeDisciplina'),
	});
	var label = Ti.UI.createLabel({
		text: disciplinas.fieldByName('NomeDisciplina'),
		id: disciplinas.fieldByName('NomeDisciplina'),
		height:'auto',
		left:'10dp',
		right: '10dp',
		top:'5dp',
		color:'#000',
		touchEnabled:false
	});

	row.add(label)
	dados.push(row);
	disciplinas.next();
}
disciplinas.close();
db.close();

// create table view
var tvDisciplinasApoio = Titanium.UI.createTableView({
	top: 0,
	height: calcularProporcaoAlturaTela(75),
	data: dados
});

// create table view event listener
tvDisciplinasApoio.addEventListener('click', function(e)
{
    var detalhes = new FormConsultaMaterialDisciplina(e.rowData.id);
	detalhes.create();

});

var btnVoltar = Ti.UI.createButton
({
	width: calcularProporcaoLarguraTela(80),
	height: calcularProporcaoAlturaTela(10),
	top: calcularProporcaoAlturaTela(80),
    left: calcularProporcaoLarguraTela(10),
```

```
    title: 'Voltar'
});

btnVoltar.addEventListener('click', function(e){
        formConsultaMaterial.close();
});

formConsultaMaterial.add(btnVoltar);
formConsultaMaterial.add(tvDisciplinasApoio);
formConsultaMaterial.open();
```

B.5 FormConsultaMaterialDisciplina.js

```
function FormConsultaMaterialDisciplina(_NomeDisciplina)
{
        this.create = function()
        {
                Ti.API.info(_NomeDisciplina);

                var dados = [];
                var formConsultaMaterialDisciplina = Titanium.UI.createWindow({
                        top: 0,
                        height: pegarAlturaTela(),
                        width: pegarLarguraTela(),
                        backgroundColor: 'white'
                });

                //Carrega as informações da tabela no vetor de dados para inserir no tableView
                var db = Ti.Database.open('MinhaUnoDB')
                var disciplinas = db.execute('SELECT publicacao, nome, descricao, url FROM MaterialApoio where NomeDisciplina = ?
                        ORDER BY Publicacao', _NomeDisciplina);
                while (disciplinas.isValidRow())
                {
                        var row = Ti.UI.createTableViewRow({
                                height: '70dp',
                                id: disciplinas.fieldByName('url'),
                        });
                        var label = Ti.UI.createLabel({
                                text: disciplinas.fieldByName('publicacao'),
                                id: disciplinas.fieldByName('publicacao'),
                                height:'auto',
                                left:'15dp',
                                top:'5dp',
                                color:'#000',
                                touchEnabled:false,
```

```
                font:
                {
        fontSize:'12dp',
        fontWeight:'bold'
}
        });

        var publicacao = Ti.UI.createLabel({
        text: disciplinas.fieldByName('nome'),
        id: disciplinas.fieldByName('descricao'),
        font:{
            fontSize:'auto'
        },
        height:'auto',
        left:'15dp',
        bottom:'5dp',
        color:'#000',
        touchEnabled:false
});

        row.add(label);
        row.add(publicacao);
        dados.push(row);
        disciplinas.next();
}
disciplinas.close();
db.close();

// create table view
var tvDisciplinasApoio = Titanium.UI.createTableView({
        top: 0,
        height: calcularProporcaoAlturaTela(75),
        data: dados
});

// create table view event listener
tvDisciplinasApoio.addEventListener('click', function(e)
{
    // event data
   Ti.Platform.openURL(e.rowData.id);

});

var btnVoltar = Ti.UI.createButton
({
        width: calcularProporcaoLarguraTela(80),
        height: calcularProporcaoAlturaTela(10),
        top: calcularProporcaoAlturaTela(80),
    left: calcularProporcaoLarguraTela(10),
    title: 'Voltar'
```

```
                });

                btnVoltar.addEventListener('click', function(e){
                        formConsultaMaterialDisciplina.close();
                });

                formConsultaMaterialDisciplina.add(btnVoltar);
                formConsultaMaterialDisciplina.add(tvDisciplinasApoio);
                formConsultaMaterialDisciplina.open();
        };
};
```

B.6 FormConsultaNota.js

```
Ti.include('Funcoes.js');
Ti.include('FormConsultaNotaDetalhe.js');

var dados = [];

var formConsultaNotas = Titanium.UI.createWindow({
        top: 0,
        height: pegarAlturaTela(),
        width: pegarLarguraTela(),
        backgroundColor: 'white'
});

//Carrega as informações da tabela no vetor de dados para inserir no tableView
var db = Ti.Database.open('MinhaUnoDB')
var disciplinas = db.execute('SELECT distinct Nome FROM NotaGraduacao ORDER BY Nome');
while (disciplinas.isValidRow())
{
        var row = Ti.UI.createTableViewRow({
                height: '60dp',
                id: disciplinas.fieldByName('Nome'),
        });
        var label = Ti.UI.createLabel({
                text: disciplinas.fieldByName('Nome'),
                id: disciplinas.fieldByName('Nome'),
                height:'auto',
                left:'10dp',
                right: '10dp',
                top:'5dp',
                color:'#000',
                touchEnabled:false
        });
```

```
        row.add(label)
        dados.push(row);
        disciplinas.next();
}
disciplinas.close();
db.close();

// create table view
var tvDisciplinasApoio = Titanium.UI.createTableView({
        top: 0,
        height: calcularProporcaoAlturaTela(75),
        data: dados
});

// create table view event listener
tvDisciplinasApoio.addEventListener('click', function(e)
{
        var db = Ti.Database.open('MinhaUnoDB')
        var detalheNotas = db.execute('select nome, notaG3, mediaFinal, notaG2, notaG1, estadoMateria, statusAcademico from
                notagraduacao where nome = ?', e.rowData.id);
        var conteudo, listaBotoes;

        if(detalheNotas.fieldByName('estadoMateria') == 'fechada')
        {
                var msgG1 = 'Média de G1: ' + detalheNotas.fieldByName('notaG1') + '\n';
                var msgG2 = 'Média de G2: ' + detalheNotas.fieldByName('notaG2') + '\n';
                var msgMediaFinal = 'Média Final: ' + detalheNotas.fieldByName('mediaFinal') + '\n';
                var statusAcademico = 'Status: ' + detalheNotas.fieldByName('statusAcademico') + '\n';

                if(detalheNotas.fieldByName('notaG3') != "")
                {
                        var msgG3 = 'Nota da G3: ' + detalheNotas.fieldByName('notaG3') + '\n';
                        conteudo = statusAcademico + msgG1 + msgG2 + msgG3 + msgMediaFinal;
                }
                else{
                        conteudo = statusAcademico + msgG1 + msgG2 + msgMediaFinal;
                }

                listaBotoes = ['Fechar']
        }
        else{
                var msgG1 = 'Média de G1: ' + detalheNotas.fieldByName('notaG1') + '\n';
                var msgG2 = 'Média de G2: ' + detalheNotas.fieldByName('notaG2') + '\n';

                conteudo = msgG1 + msgG2;
                listaBotoes = ['Fechar', 'Detalhes'];
        }

        var Msg = Ti.UI.createAlertDialog
        ({
```

```
            title: e.rowData.id,
            message: conteudo,
            buttonNames: listaBotoes
        });

        Msg.addEventListener('click', function(click){
                if(click.index == 1)
                {
                        var detalhes = new FormConsultaNotaDetalhe();
                        detalhes.create(e.rowData.id);
                }
        });

        Msg.show();
        detalheNotas.close();
        db.close();
});

var btnVoltar = Ti.UI.createButton
({
        width: calcularProporcaoLarguraTela(80),
        height: calcularProporcaoAlturaTela(10),
        top: calcularProporcaoAlturaTela(80),
    left: calcularProporcaoLarguraTela(10),
    title: 'Voltar'
});

btnVoltar.addEventListener('click', function(e){
        formConsultaNotas.close();
});

formConsultaNotas.add(btnVoltar);
formConsultaNotas.add(tvDisciplinasApoio);
formConsultaNotas.open();
```

B.7 FormConsultaNotaDetalhe.js

```
function FormConsultaNotaDetalhe()
{
        this.create = function(_NomeDisciplina)
        {
                var dados = [];
                var formConsultaNotaDetalhe = Titanium.UI.createWindow({
                        top: 0,
                        height: pegarAlturaTela(),
                        width: pegarLarguraTela(),
```

```
        backgroundColor: 'white'
});

//Carrega as informações da tabela no vetor de dados para inserir no tableView
var db = Ti.Database.open('MinhaUnoDB')
var avaliacoes = db.execute('SELECT nome, peso, nota, data FROM NotaGraduacaoAvaliacao where NomeDisciplina = ?
    ORDER BY data', _NomeDisciplina);

if(!avaliacoes.rowCount)
{
        msgBox('Atenção', 'Não existem avaliações cadastradas!');
        avaliacoes.close();
        db.close();
        return;
}

while (avaliacoes.isValidRow())
{
        var texto;
        if(avaliacoes.fieldByName('nota') != ""){
                texto = 'Nota: ' + avaliacoes.fieldByName('nota') + '\n' + 'Peso: ' + avaliacoes.fieldByName('peso') + '
                        \n' + 'Data: ' + avaliacoes.fieldByName('data') ;
        }
        else{
                texto = 'Peso: ' + avaliacoes.fieldByName('peso') + '\n' + 'Data: ' + avaliacoes.fieldByName('data') ;
        }
        var row = Ti.UI.createTableViewRow({
                height: '90dp',
        });
        var label = Ti.UI.createLabel({
                text: avaliacoes.fieldByName('nome'),
                height:'auto',
                left:'15dp',
                top:'5dp',
                color:'#000',
                touchEnabled:false,

                font:
                {
        fontSize:'15dp',
        fontWeight:'bold'
}
        });

        var detalhes = Ti.UI.createLabel({
        text: texto,
        font:{
            fontSize:'auto'
        },
        height:'auto',
```

```
                left:'15dp',
                bottom:'5dp',
                color:'#000',
                touchEnabled:false
            });

                row.add(label);
                row.add(detalhes);
                dados.push(row);
                avaliacoes.next();
            }
            avaliacoes.close();
            db.close();

            // create table view
            var tvAvaliacoes = Titanium.UI.createTableView({
                top: 0,
                height: calcularProporcaoAlturaTela(75),
                data: dados
            });

            var btnVoltar = Ti.UI.createButton
            ({
                width: calcularProporcaoLarguraTela(80),
                height: calcularProporcaoAlturaTela(10),
                top: calcularProporcaoAlturaTela(80),
              left: calcularProporcaoLarguraTela(10),
              title: 'Voltar'
            });

            btnVoltar.addEventListener('click', function(e){
                formConsultaNotaDetalhe.close();
            });

            formConsultaNotaDetalhe.add(btnVoltar);
            formConsultaNotaDetalhe.add(tvAvaliacoes);
            formConsultaNotaDetalhe.open();
        };
}
```

B.8 FormConsultaHorarioSemestre.js

```
Ti.include('Funcoes.js');
Ti.include('FormConsultaHorarioDetalhe.js');

var dados = [];
```

```
var formHorariosSemestre = Titanium.UI.createWindow({
        top: 0,
        height: pegarAlturaTela(),
        width: pegarLarguraTela(),
        backgroundColor: 'white'
});

//Carrega as informações da tabela no vetor de dados para inserir no tableView
var db = Ti.Database.open('MinhaUnoDB')
var disciplinas = db.execute('SELECT distinct Nome, Codigo FROM HorarioSemestre ORDER BY Nome');
while (disciplinas.isValidRow())
{
        var row = Ti.UI.createTableViewRow({
                height: '60dp',
                id: disciplinas.fieldByName('codigo'),
        });
        var label = Ti.UI.createLabel({
                text: disciplinas.fieldByName('Nome'),
                height:'auto',
                left:'10dp',
                right: '10dp',
                top:'5dp',
                color:'#000',
                touchEnabled:false
        });

        row.add(label)
        dados.push(row);
        disciplinas.next();
}
disciplinas.close();
db.close();

// create table view
var tvHorariosSemestre = Titanium.UI.createTableView({
        top: 0,
        height: calcularProporcaoAlturaTela(75),
        data: dados
});

// create table view event listener
tvHorariosSemestre.addEventListener('click', function(e)
{
        var db = Ti.Database.open('MinhaUnoDB')
        var detalheDisciplina = db.execute('select nome, turma, curso, dataG2, dataG3, professor, creditos, turno, grade, periodo from
                HorarioSemestre where codigo = ?', e.rowData.id);
        var conteudo, listaBotoes;

        var msgTurma = (detalheDisciplina.fieldByName('turma') != null) ? 'Turma: ' + detalheDisciplina.fieldByName('turma') + '\n' : ''
                ;
```

```
        var msgCurso = (detalheDisciplina.fieldByName('curso') != null) ? 'Curso: ' + detalheDisciplina.fieldByName('curso') + '\n' : '';
        var msgDataG2 = (detalheDisciplina.fieldByName('dataG2') != null) ? 'Data da G2: ' + detalheDisciplina.fieldByName('dataG2')
            + '\n' : '';
        var msgDataG3 = (detalheDisciplina.fieldByName('dataG3') != null) ? 'Data da G3: ' + detalheDisciplina.fieldByName('dataG3')
            + '\n' : '';
        var msgProfessor = (detalheDisciplina.fieldByName('professor') != null) ? 'Professor: ' + detalheDisciplina.fieldByName('
            professor') + '\n' : '';
        var msgTurno = (detalheDisciplina.fieldByName('turno') != null) ? 'Turno: ' + detalheDisciplina.fieldByName('turno') + '\n' : '';
        var msgGrade = (detalheDisciplina.fieldByName('grade') != null) ? 'Grade: ' + detalheDisciplina.fieldByName('grade') + '\n' : '';
        var msgPeriodo = (detalheDisciplina.fieldByName('periodo') != null) ? 'Período: ' + detalheDisciplina.fieldByName('periodo') + '
            \n' : '';

        conteudo = msgTurma + msgCurso + msgGrade + msgTurno + msgProfessor + msgDataG2 + msgDataG3;

        listaBotoes = ['Fechar', 'Detalhes'];


        var Msg = Ti.UI.createAlertDialog
        ({
            title: detalheDisciplina.fieldByName('nome'),
            message: conteudo,
            buttonNames: listaBotoes
        });

        Msg.addEventListener('click', function(click){
                if(click.index == 1)
                {
                        var detalhes = new FormConsultaHorarioDetalhe(e.rowData.id);
                        detalhes.create();
                }
        });

        Msg.show();
        detalheDisciplina.close();
        db.close();
});

var btnVoltar = Ti.UI.createButton
({
        width: calcularProporcaoLarguraTela(80),
        height: calcularProporcaoAlturaTela(10),
        top: calcularProporcaoAlturaTela(80),
    left: calcularProporcaoLarguraTela(10),
    title: 'Voltar'
});

btnVoltar.addEventListener('click', function(e){
        formHorariosSemestre.close();
});
```

```
formHorariosSemestre.add(btnVoltar);
formHorariosSemestre.add(tvHorariosSemestre);
formHorariosSemestre.open();
```

B.9 FormConsultaHorarioDetalhe.js

```
function FormConsultaHorarioDetalhe(_CodigoDisciplina) {
    this.create = function () {
        var dados = [];

        var formConsultaHorarioDetalhe = Titanium.UI.createWindow({
            top: 0,
            height: pegarAlturaTela(),
            width: pegarLarguraTela(),
            backgroundColor: 'white'
        });

        /*
         * Carrega as informações da tabela no vetor
         * de dados para inserir no tableView
         */
        var db = Ti.Database.open('MinhaUnoDB')
        var horarios = db.execute('SELECT hora, diasemana, data, ocorreu FROM \
                HorarioSemestreDisciplina where codigo = ?', _CodigoDisciplina);

        if (!horarios.rowCount) {
            msgBox('Atenção', 'Não existem horários cadastrados para esta\
                    disciplina!');
            horarios.close();
            db.close();
            return;
        }

        while (horarios.isValidRow()) {
            if (horarios.fieldByName('data') && horarios.fieldByName('hora')
                    && horarios.fieldByName('diasemana'))
            {
                var texto = 'Data: ' + horarios.fieldByName('data') + ' - '
                        + 'Hora: ' + horarios.fieldByName('hora') + '\n'
                        + 'Dia da Semana: ' + horarios.fieldByName('diasemana')
                        + '\n';
                var estilo = (horarios.fieldByName('ocorreu') == 'false')
                        ? 'bold' : 'normal';
                var registro = Ti.UI.createTableViewRow({
                    height: '60dp'
                });
```

```
                var descricao = Ti.UI.createLabel({
                    text: texto,
                    font: {
                        fontWeight: estilo
                    }
                });

                registro.add(descricao);
                dados.push(registro);
            }
            horarios.next();
        }

        var tvHorarios = Titanium.UI.createTableView({
            top: 0,
            height: calcularProporcaoAlturaTela(75),
            data: dados
        });

        var btnVoltar = Ti.UI.createButton({
            width: calcularProporcaoLarguraTela(80),
            height: calcularProporcaoAlturaTela(10),
            top: calcularProporcaoAlturaTela(80),
            left: calcularProporcaoLarguraTela(10),
            title: 'Voltar'
        });

        btnVoltar.addEventListener('click', function (e) {
            formConsultaHorarioDetalhe.close();
        });

        formConsultaHorarioDetalhe.add(tvHorarios);
        formConsultaHorarioDetalhe.add(btnVoltar);
        formConsultaHorarioDetalhe.open();
    };
}
```

B.10 Funcoes.js

```
function pegarLarguraTela()
{
        return Ti.Platform.displayCaps.platformWidth;
}

function pegarAlturaTela()
```

```
{
        return Ti.Platform.displayCaps.platformHeight;
}

//Retorna a quantidade de pixels da Altura da tela representados pela porcentagem passada como parâmetro
function calcularProporcaoAlturaTela(Porcentagem)
{
        return (pegarAlturaTela() * (Porcentagem / 100));
}

//Retorna a quantidade de pixels da Largura da tela representados pela porcentagem passada como parâmetro
function calcularProporcaoLarguraTela(Porcentagem)
{
        return (pegarLarguraTela() * (Porcentagem / 100));
}

function msgBox(Titulo, Mensagem)
{
        var Msg = Ti.UI.createAlertDialog
        ({
            title: Titulo,
            message: Mensagem,
            buttonNames: ['Ok']
        });

        Msg.show();
}
```

B.11 app.js

```
// this sets the background color of the master UIView (when there are no windows/tab groups on it)
Ti.include('FormLogin.js');

Titanium.UI.setBackgroundColor('#000');

var login = new FormLogin();
login.create();
```

APÊNDICE C – Icebreak REST Server

Neste capítulo é apresentada a implementação do Icebreak REST Server. A implementação apresentada é similar a original, porém houve a necessidade de ser removido o tamanho do arquivo retornado no cabeçalho do HTML, o que pode ser visto na função *sendResponse*. Foi destacada nesta função a linha comentada.

```
/* */
/* Copyright [2010] [System & Method A/S] */
/* */
/* Licensed under the Apache License, Version 2.0 (the "License"); */
/* you may not use this file except in compliance with the License. */
/* You may obtain a copy of the License at */
/* */
/* http://www.apache.org/licenses/LICENSE-2.0 */
/* */
/* Unless required by applicable law or agreed to in writing, software */
/* distributed under the License is distributed on an "AS IS" BASIS, */
/* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. */
/* See the License for the specific language governing permissions and */
/* limitations under the License. */
/* */
/* Design - Niels Liisberg */
/* */

package edu.unochapeco.saa;

import java.io.*;
import java.net.*;
import java.util.*;
import java.text.*;
import java.io.FileInputStream;
import java.io.IOException;
import java.util.Properties;
import java.net.URL.*;
import java.net.URLDecoder;
```

```
/**
 * Super tiny HTTP serverside protocol for monolitic RESTservice applications
 * Simply drop the IceBreakRestServer jar file in your project ( classpath) and you are golden.
 *
 * A simple server looks like this:
 *
 *
 * <pre>
 * {@code
 * // Drop this jar-file into you project
 * import IceBreakRestServer.*;
 * import java.io.IOException;
 * public class Simple {
 *
 * public static void main(String[] args) {
 *
 * // Declare the IceBreak HTTP REST server class
 * IceBreakRestServer rest;
 *
 *
 * try {
 *
 * // Instantiate it once
 * rest = new IceBreakRestServer();
 *
 * while (true) {
 *
 * // Now wait for any HTTP request
 * // the "config.properties" file contains the port we are listening on
 * rest.getHttpRequest();
 *
 * // If we reach this point, we have received a request
 * // now we can pull out the parameters from the query-string
 * // if not found we return the default "N/A"
 * String name = rest.getQuery("name", "N/A");
 *
 * // we can now produce the response back to the client.
 * // That might be XML, HTML, JSON or just plain text like here:
 * rest.write("Hello world - the 'name' parameter is: " + name );
 * }
 * } catch (IOException ex) {
 * System.out.println(ex.getMessage());
 * }
 * }
 * }
 * }
 * </pre>
 */
```

```
public class RestServer {

  private ServerSocket providerSocket = null;
  private Socket connection = null;
  private PrintWriter pw;
  private String ContentType;
  private String Status;
  private StringBuilder resp = new StringBuilder(2048);
  private Boolean doFlush = false;
  private int Port ;
  private int Queue ;
  private InputStream in;

  /** This is the complete querysting including the resource. Just as you write it in your browser - you have to URL decode it or rather use
      getQuery to get paramter */
  public String request;
  /** This is the contents sent by a POST */
  public String payload;
  /** This is the request type GET, POST, HEAD - your application have to responde coretly to this ( ore simply ignore it */
  public String method ;
  /** This is the complete querysting after the resource as you write it in your browser - you have to URL decode it or rather use getQuery
      to get paramter */
  public String queryStr;
  /** This is the name of the resource to run or get i.e. http://x/myApp.aspx/p1=abc it will return /myApp.aspx */
  public String resource;
  /** This is the version of the HTTP protocol requested */
  public String httpVer;
  /** Set this to true to get some system.out.print */
  public Boolean debug = false;

  /** This is the HTTP headers in the request. Use normal "Map" methods */
  public Map<String, String> header = new HashMap<String, String>();
  /** This is the HTTP quesystring parameters as map. Use normal "Map" methods or getQuery() method */
  public Map<String, String> parms = new HashMap<String, String>();

  private void loadProps ( ) {
    Properties prop = new Properties();

    try {
      //load a properties file
      prop.load(new FileInputStream("config.properties"));
      Port = Integer.parseInt(prop.getProperty("restserver.port","65000"));
      Queue = Integer.parseInt(prop.getProperty("restserver.queuesize", "10"));
    } catch (IOException ex) {
      // ex.printStackTrace();
    }
  }
  /**
        * Contructor, returns an instance of the rest server
        */
```

```
public RestServer() {
  loadProps ();
}

/**
        * Set the contents type of the HTTP contens. It has to conform
 * the mime type. By default it has the value of "text/plain; charset=utf-8"
        * @param contents type string to set
        */
public void setContentType(String s) {
  ContentType = s;
}
/**
        * Set the status of HTTP contens.
 * @see <a href="http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html">HTTP status codes</a>
        * @param status string . by default the is "200 OK"
        */
public void setStatus(String s) {
  Status = s;
}

/**
        * Set the TCP/IP port that you server is listening on. This is by default port 65000 and you can
 * set this value in the config.prperties file. Or you can set it programatically here but before issuing
 * a "getRequest()".
 * @param port TCP/IP port to listen on
        */
public void setPort(int port) {
  Port = port;
}

/**
        * Set the TCP/IP queue depth for your HTTP server . This is by default port 10 and you can
 * set this value in the config.prperties file. Or you can set it programatically here but before issuing
 * a "getRequest()".
 * @param port TCP/IP port to listen on
        */
public void setQueue(int queue) {
    Queue = queue;
}

/**
        * Returns the parameter from the querystring with the name of "key". if the querystring
 * parameter was not fount it will return the default paramter
 * Note: key is case sensitive!!
 * @param Key - to return value for in the querystring
 * @param Default - when key is not found this wil be the default value
 * @return value of the querystring parameter
        */
public String getQuery(String Key , String Default) {
```

```
    String temp = parms.get(Key);
    if (temp == null) return Default;
    return temp;
}

/**
        * Returns the parameter from the querystring with the name of "key". if the querystring
 * parameter was not fount it will <code>null</code>
 * Note: key is case sensitive!!
 * @param Key - to return value for in the querystring
 * @return value of the querystring parameter
        */
public String getQuery(String Key) {
    return parms.get(Key);
}

/**
        * Just return a simple string with current timestamp in hh:mm:ss format
 * @return current time in hh:mm:ss format
        */
public String now () {
   String s;
   Format formatter;
   Date date = new Date();
   formatter = new SimpleDateFormat("hh:mm:ss");
   s = formatter.format(date);
   return s;
}

private static Map<String, String> getQueryMap(String query) {
    String[] params = query.split("&");
    Map<String, String> map = new HashMap<String, String>();
    for (String param : params) {
       int p = param.indexOf('=');
       if (p >= 0) {
          String name = param.substring( 0, p);
          String value = param.substring( p+1);
          String s = URLDecoder.decode(value);
          map.put(name, s);
       }
    }
    return map;
}

// This handles both windows <CR><LF> and mac/aix/linux <CR>
// and returns both end of header and end of line sequence
private int isEol(byte [] buf , int i) {
   if (buf[i] == 0x0d &&buf[i+1] == 0x0a) {
      if (buf[i+2] == 0x0d &&buf[i+3] == 0x0a) {
         return -4; // End Of header
```

```
      }
      return 2;
    }
    if (buf[i] == 0x0d ) {
      if (buf[i+1] == 0x0d ) {
        return −2; // End Of header
      }
      return 1;
    }
    if (buf[i] == 0x0a ) {
      if (buf[i+1] == 0x0a ) {
        return −2; // End Of header
      }
      return 1;
    }
    return 0;
}


 private void unpackRequest() throws IOException {

    byte buf [] = new byte[32768];
    in = connection.getInputStream();
    int read = in.read(buf);
    int len =0, pos =0, eol=0;
    header.clear();
    parms.clear();
    request = payload = method = queryStr = httpVer = resource = null;
    for (int i = 0; i < read && eol >= 0; i++) {
        eol = isEol(buf , i);
        if (eol > 0) {
          // First line is the request. Now parse that partial
          if (request == null) {
              request = new String(buf, pos , len);
              String [] temp = request.split(" ");
              method = temp[0];
              queryStr = temp[1];
              httpVer = temp[2];
              int p = queryStr.indexOf('?');
              if (p>=0) {
                resource = queryStr.substring( 0, p);
                parms = getQueryMap(queryStr.substring( p+1));
              } else {
                resource = queryStr;
              }
          // Following lines are the header − put them into a map
          } else {
              String param = new String(buf, pos , len);
              int p = param.indexOf(':');
              String name = param.substring( 0, p);
```

```
            String value = param.substring( p+1);
            header.put(name, value.trim());
          }
          len = 0;
          pos = i + eol;
          i+=eol−1;
        } else if (eol < 0) {
          pos = i + (−eol);
          payload = new String(buf, pos , read − pos);
        } else {
          len ++;
        }
    }

    // this is only for debugging
    if (debug) {
      System.out.println(”resource: ” + request);
      System.out.println(”method: ” + method);
      System.out.println(”resource: ” + resource);
      System.out.println(”queryStr: ” + queryStr);
      System.out.println(”httpVer: ” + httpVer);
      System.out.println(”header : ” + header );
      System.out.println(”parms : ” + parms );
    }

}

private void sendResponse () {

    pw.print(”HTTP/1.1 ” + Status + ”\r\n” +
            ”Connection: Keep−Alive\r\n” +
            ”Accept: multipart/form−data\r\n”+
            ”Accept−Encoding: multipart/form−data\r\n” +
            ”Server: IceBreak Java Services\r\n” +
            ”cache−control: no−store\r\n” +
            // LINHA COMENTADA POIS OCASIONAVA UM PROBLEMA DE FALTA DE
            // DADOS NO JSON DE RETORNO
            //”Content−Length: ” + Integer.toString(resp.length()) + ”\r\n” +
            ”Content−Type: ” + ContentType + ”\r\n” +
            ”\r\n” + resp.toString());
    pw.flush();
}

/**
        ∗ This waits for the next HTTP request from the client
 *
        ∗/
public void getHttpRequest () throws IOException {
    if (providerSocket == null) {
      providerSocket = new ServerSocket(Port , Queue);
```

```
    }
    if (doFlush) flush();

    connection = providerSocket.accept();
    pw = new PrintWriter(connection.getOutputStream());
    resp.setLength(0);
    unpackRequest();
    ContentType = "text/plain ;charset=utf-8";
    Status = "200 OK";
    doFlush = true;
  }

  /**
        * write a string back a tring to the client. The complete result will be
   * send back to the client when you issue a "flush" or do the next "getHttpRequest()"
   * @param String - to send back to the client
        */
  public void write(String s) {
    resp.append(s);
  }

  /**
        * send back the complete response to the client. Now we are ready to wait for the next request by issue a "getHttpRequest()"
        */
  public void flush() throws IOException {
    sendResponse ();
    connection.close();
    doFlush = false;
  }
}
```