# *Implementação do Módulo Web Baseado na Arquitetura Cliente-Servidor para o Aplicativo Móvel Educacional e Open-Source Sem!o*

Uma Ferramenta Educacional de Apoio ao Ensino-Aprendizagem da Semiologia Médica

Gabriele Bueno de Oliveira, Jeziel Mateus de Abreu
Universidade Federal da Integração Latino-Americana
UNILA
Foz do Iguaçu, PR, Brasil
{gb.oliveira.2017, jmd.abreu.2016}@aluno.unila.edu.br

Joylan Nunes Maciel, Marcelo Nepomoceno Kapp
Universidade Federal da Integração Latino-Americana
UNILA
Foz do Iguaçu, PR, Brasil
{joylan.maciel, marcelo.kapp}@unila.edu.br

***Resumo*—O uso das tecnologias de informação na saúde tem promovido contribuições positivas no ensino-aprendizagem da medicina. Este trabalho apresenta o processo de desenvolvimento de um módulo web e sua integração ao aplicativo móvel educacional e *open-source Sem!o*, o qual é utilizado no processo de ensino-aprendizagem da semiologia médica. O módulo web está em fase de implementação e emprega a arquitetura Cliente-Servidor, o padrão arquitetural REST, as ferramentas da plataforma Google Android e o modelo de desenvolvimento de software do Processo Unificado. A construção do módulo web permitirá o uso do aplicativo *Sem!o* por diversos usuários simultaneamente, via Internet, agregando características de interatividade e estimulando um melhor aprendizado dos termos médicos para os estudantes de medicina. Isto contribuirá positivamente na qualificação dos futuros profissionais médicos, pois o correto significado dos termos médicos promove diagnósticos mais precisos e efetivos para a sociedade. As características de flexibilidade de acesso e atualização da base de dados do aplicativo *Sem!o* são diferenciadas e o tornam aplicável em qualquer domínio do conhecimento na educação.**

***Keywords: Ensino-Aprendizagem; Android; REST; Aplicativo Web; Semiologia Médica.***

## I. INTRODUÇÃO

A geração de dispositivos móveis atuais permite que os usuários realizem diversas tarefas simultaneamente, antes realizadas por computadores *desktop* conectados ou não à Internet. Podem-se destacar como principais características da computação móvel a mobilidade e a flexibilidade que engloba também portabilidade, usabilidade, popularidade e conectividade desses dispositivos [1].

O uso de técnicas computacionais como instrumento de apoio à medicina tem gerado contribuições importantes e positivas, entre as quais, intercâmbio de informações médicas, aprimoramento de serviços e diagnósticos, a eficiência na tomada de decisão quanto à escolha da terapêutica realizada, aperfeiçoamento da educação médica, inclusive à distância [2,3,4].

Sob a perspectiva do ensino-aprendizagem da medicina, o uso das tecnologias de informação tem promovido grandes contribuições positivas na prática do ensino e da pesquisa [5,6,7,8]. Mais especificamente sobre o aprendizado e conhecimento de termos médicos, e conceitos relacionados a estes termos, diversas soluções computacionais estão disponíveis para acesso e utilização, tais como: o Sheppard Software [9] que consiste em um jogo para o aprendizado e melhora do vocabulário médico em inglês; o site Sporcle [10] que permite a criação de jogos de perguntas sobre termos médicos em inglês; o aplicativo (*app*) Medical Terminology Quiz [11] e Learn Medical Teraminology [12], os quais permitem a consulta e aprendizado de termos médicos em inglês; os aplicativos Terminologia Médica Free [13] e Dicionário de Saúde Offline [14] que permitem a consulta de termos médicos em português, tal como um dicionário ou glossário, dentre outras aplicações

Ainda nesse cenário, destacam-se dois aplicativos: o SemioQuiz [15] que fornece um Quiz sobre a semiologia médica e o Kahoot [16], um *app* interativo que permite ao docente cadastrar questões individuais e realizar campeonatos de quiz em sala de aula, em qualquer idioma. Alguns dos *apps* listados anteriormente permitem realizar jogos de pergunta e respostas (quizzes) e promovem maior interatividade do usuário durante o processo de aprendizagem. Todavia, a maioria destes aplicativos são para a língua inglesa ou não são livres, restringindo o acesso gratuito a todas as funcionalidades e, principalmente, ao banco de dados do *app*. No caso do Kahoot, apesar da interatividade promovida e de aceitar diversos idiomas, falta flexibilidade durante o cadastramento das questões que deve ser feito de modo individual e manual, exigindo tempo e dedicação dos docentes e gestores. Além disso, o quiz em grupo do Kahoot funciona somente online via Internet.

Nesse sentido, este trabalho tem o objetivo geral de projetar e desenvolver o módulo de funcionamento web, baseado na arquitetura Cliente-Servidor [17], para o aplicativo móvel educacional *Sem!o* [18], um software livre que está sendo desenvolvido pelo Núcleo de Informática em Ciências da Saúde[1] da Universidade Federal da Integração Latino-Americana (UNILA). Aplicado inicialmente na educação médica, os principais diferenciais do *Sem!o* consistem na flexibilidade de gestão e atualização do banco de dados de questões [19], na interatividade promovida com sua utilização e possibilidade de expansão e aplicação em outros domínios do conhecimento. Isto pode contribuir para melhorar o processo de ensino-aprendizagem educacional em qualquer área da educação. Desse modo, os objetivos

1 Informações adicionais disponíveis em < http://nics.tk >.

específicos consistem em: 1) avaliar as tecnologias e métodos de desenvolvimento de aplicativos móveis em Android atuais baseadas na arquitetura Cliente-Servidor; 2) implementar o servidor (*back-end*) para tratamento das requisições e respostas para o banco de dados; e 3) desenvolver o cliente (*front-end*) para o sistema operacional móvel Google Android, permitindo ao usuário manipular dos dados por meio da interface gráfica interativa. Portanto, este trabalho está organizado do seguinte modo: a seção II apresenta as ferramentas, materiais e métodos empregados no desenvolvimento, a seção III exibe os resultados alcançados até o momento, e a seção IV as considerações finais e trabalhos futuros.

## II. MATERIAIS E MÉTODO

Considerando os objetivos definidos na Seção I, um estudo bibliográfico foi realizado acerca das tecnologias, ferramentas e metodologias de desenvolvimento *open-source*, as quais são detalhadas a seguir.

### *A. Tecnologias e Ferramentas*

*a)* Linguagem PHP**:** A linguagem de programação PHP surgiu em 1994 e desde então vem sendo aprimorada e utilizada para desenvolvimento web. Possui vantagem em relação a outras linguagens que têm o mesmo propósito por ser multiplataforma, dinâmica e gratuita. Além disso, é uma linguagem de execução robusta, sólida e estável em relação a implementação e quantidade de recursos disponíveis [20]. Os códigos PHP são executados no servidor (*back-end*), cuja função é responder as requisições, executar scripts de processamento e enviar os resultados para o cliente. Portanto, a linguagem PHP foi empregada na implementação *back-end* do *app* proposto.

*b)* Servidor de banco de dados MySQL: A principal função dos bancos de dados é o armazenamento das informações manipuladas no *app*. Geralmente utiliza um modelo de representação relacional dos dados, ou seja, um grupo de registros formando uma ou várias tabelas que facilitam a organização e a comunicação entre os dados em documentos ou softwares [21]. A gestão do banco de dados é realizada por Sistemas de Gerenciamento de Banco de Dados (SGBD), responsáveis por disponibilizar uma interface para que programas e usuários externos acessem o banco de dados, realizem backups, consultas e manipulação com controle de acesso à informações. Neste trabalho utilizou-se o SGBD MySQL por ser gratuito, estável e atender aos requisitos do *app* proposto.

*c)* Servidor de Aplicação Apache e XAMPP[2]: Empregou-se neste trabalho o Apache como servidor de aplicação, sendo que este é o servidor web mais utilizado no mundo e trabalha com uma grande variedade de ambientes e plataformas [17]. É responsável pela gestão do protocolo *Hypertext Transfer Protocol (*HTTP) e documentos *Hypertext Markup Language (*HTML), disponibilizando arquivos no formato de hipertexto e realizando a conexão entre os navegadores (*web browsers*) do usuário e o servidor. Além disso, na construção da solução utilizou-se a ferramenta de desenvolvimento *open-source* XAMPP[3], que agrega o banco de dados, o servidor de aplicação Apache e o ambiente de desenvolvimento necessário ao projeto. O XAMPP é gratuito e de fácil instalação, viabiliza a construção de aplicações web provendo uma comunicação efetiva entre o cliente e servidor.

*d)* Padrão arquitetural REST e Framework SLIM: Aplicou-se o modelo arquitetural *Representational State Transfer* (REST) proposto em [22], cujo padrão é baseado no conceito de recursos e emprega os métodos GET, PUT, POST, DELETE e etc, do protocolo HTTP para a realização e tratamento das requisições e respostas em um sistema web Cliente-Servidor. Tais métodos não manipulam os recursos, mas a representação de seus estados [22]. Os serviços que seguem o padrão arquitetural REST são denominados RESTful e provêem maior facilidade para o desenvolvimento de aplicativos móveis, uma vez que a praticidade é incorporada por conta da demanda de restrições que os equipamentos impõem [23]. A implementação REST foi elaborada por meio do *Slim Framework*[4], cuja função consiste em facilitar a criação de rotas e *Uniform Resource Locator* (URL), atuando principalmente na abstração dos controladores presentes no servidor. Estes controladores recebem as requisições HTTP dos clientes, as processam de acordo com a rota (URL) utilizada e retornam as respostas [17].

*e)* Linguagem Java e Plataforma Android Studio: Java é uma linguagem de programação orientada a objetos e utilizada em uma vasta gama de plataformas e ambientes heterogêneos. Provê compatibilidade, segurança e foi a primeira linguagem nativa utilizada no desenvolvimento de aplicativos para Android [24]. A plataforma oficial da Google para desenvolvimento de *app* nativos Android é o ambiente de programação Android Studio[5], que permite empregar a linguagem Java e a edição inteligente de códigos contendo diversos recursos visuais que facilitam a construção da interface gráfica do usuário [25].

*f)* Biblioteca Retrofit: É uma biblioteca de recursos *open-source* que intermedia a comunicação web entre *back-end* e *front-end* [26], transformando a comunicação HTTP em uma interface de programação Java. Utiliza anotações HTTP em Java que são responsáveis por fornecer o método de requisição e sua respectiva URL. Também permite serializar os dados por meio de um conversor denominado GSON, trabalhando com chamadas (*calls*) que fazem requisições (síncronas ou assíncronas) ao servidor *back-end* [26].

### *B. Método de Desenvolvimento*

O módulo web do *Sem!o* foi projetado com base na arquitetura Cliente-Servidor [17], em que o servidor trata e responde as requisições, via protocolo HTTP, que são realizadas pelos clientes, neste caso *app* Android. Esta

2 Mais detalhes disponíveis em < https://www.apachefriends.org >.

3 Detalhes disponíveis em < https://www.apachefriends.org/blog/news-article-61070.html >.

4 Detalhes disponíveis em < http://www.slimframework.com >.

5 Detalhes disponíveis em < https://developer.android.com/docs >.

arquitetura simplifica o desenvolvimento e a manutenção, sendo seus componentes detalhados a seguir:

a) Servidor (*back-end*)**:** O servidor desenvolvido em PHP realiza a comunicação entre o banco de dados e o tratamento de requisições HTTP, executando manipulações de inserção, consultas, edições e exclusões no banco de dados com base nos métodos HTTP (POST, GET, DELETE, etc). Toda requisição válida recebida pelo servidor gera uma resposta ao cliente solicitante. Após a consulta ao banco de dados o servidor retorna uma resposta de sucesso ou falha. Essas respostas, quando necessário, contêm dados encapsulados no formato JSON.

Na Figura 1 é demonstrado o código PHP de tratamento do servidor para tratar uma requisição de consulta de Usuário e seus atributos. Observa-se o método HTTP GET utilizado para a URL (*end-point) /consultarUsuarios*, a chamada do serviço de consulta de usuários no banco de dados *$db->consultarUsuarios()* e a resposta enviada para o cliente no formato JSON *json_encode($response_data)*.

```
$app->get('/consultarUsuarios', function(Request $request,
    Response $response){
  $db = new DbOperations;
  $users = $db->consultarUsuarios();
  $response_data = array();
  $response_data['error'] = false;
  $response_data['users'] = $users;
  $response->write(json_encode($response_data));
  return $response->withHeader('Content-type',
    'application/json')->withStatus(200);
});
```

Figura 1. Tratamento de uma requisição no servidor (*back-end*).

b) Cliente (*front-end*): A aplicação cliente foi desenvolvida para o sistema operacional Google Android, por meio da linguagem Java e biblioteca Retrofit, cuja classe essencial para funcionamento da comunicação de dados entre o cliente e servidor é relativa ao objeto *Service* e sua respectiva interface. A Figura 2 representa a realização de uma requisição de cadastro de usuário pelo cliente, na qual observa-se a requisição com o método POST e seu *end-point*, a chamada do serviço ao servidor *"cadastroUsuario"* e o atributos *@Field* do usuário a ser cadastrado.

```
@FormUrlEncoded
@POST("cadastroUsuario")
call<DefaultResponse> cadastroUsuario(
        @Field("nome") String nome,
        @Field("email") String email,
        @Field("senha") String senha,
        @Field("perfil") String perfil
);
```

Figura 2. Requisição do cliente (*front-end*) com a biblioteca Retrofit.

A Figura 3 apresenta o tratamento da resposta recebida no cliente para a requisição de cadastro de usuário da Figura 2. Observa-se que o objeto da chamada *call* trata os dados enviado pelo servidor ao cliente, ou seja, a resposta relativa ao serviço *cadastroUsuario(...)*. Essa resposta tratada no método *OnResponse(...)* pode ser de usuário criado (código 201) ou usuário existente (código 422), ou ainda alguma falha tratada por meio do método *onFailure(...)*.

```
Call<DefaultResponse> call = RetrofitClient.getInstance().
    getApi().cadastroUsuario(nome, email, senha, perfil);
call.enqueue(new Callback<DefaultResponse>() {
@Override
 public void onResponse(Call<DefaultResponse>call,
   Response<DefaultResponse> response) {
     if (response.code() == 201)
       // Mensagem de Usuário Criado!
     else if (response.code() == 422)
       // Mensagem de Usuário Existente!
 }
 @Override
 public void onFailure(Call<DefaultResponse>call,Throwable
   t){
     Toast.makeText(MainActivity.this, t.getMessage(),
     Toast.LENGTH_LONG).show();
    }
});
```

Figura 3. Tratamento de uma resposta no cliente (front-end) por meio da biblioteca Retrofit.

De modo similar as operações de criação, listagem, alteração e remoção (CRUD) foram desenvolvidas para os os usuários do sistema por meio do módulo web. O desenvolvimento empregou a biblioteca Retrofit e seguiu o modelo de processo de software Processo Unificado [27].

## III. RESULTADOS

As principais funcionalidades não web e existentes no aplicativo *Sem!o* estão representadas no Diagrama de Casos de Uso [27] exibido na Figura 4. Até o momento, a integração ao módulo web foi desenvolvida, está testada e funcional para os casos de uso Gerenciar Usuários e Gerenciar Categorias, conforme identificado na Figura 4.

Figura 4: Diagrama de Casos de Uso do Sem!o. Adaptado de [28].

Na Figura 5 são apresentadas exemplos de telas referentes a interface do usuário no cliente do aplicativo *Sem!o*. Na esquerda é exibida a tela inicial do sistema para login dos usuários no *app*, no centro a interface de listagem dos usuários já cadastrados no servidor da aplicação, e à direita é apresentada a tela de edição das informações de usuários, as quais são posteriormente persistidas no servidor.

Figura 5. Interfaces do módulo web (*front-end*) do *app Sem!o*: tela de login (esquerda), listagem (centro) e edição de usuário (direita).

## IV. CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

Este trabalho apresentou o processo de desenvolvimento e integração de um módulo web, baseado na arquitetura Cliente-Servidor e modelo arquitetural REST, ao *app* móvel, educacional e *open-source Sem!o*. A implementação do módulo web já foi realizada para os casos de uso Gerenciar Usuários e Categorias. Os resultados demonstram a viabilidade dos materiais e métodos empregados, validando a solução adotada.

Desenvolver todas as funcionalidades do *Sem!io* no modelo web possibilitará o uso simultâneo do *app* por vários usuários, permitindo o acesso a competições online de quizzes e aumentando a interatividade entre os participantes. Isto promoverá o estímulo acadêmico quanto ao aprendizado de termos médicos, de forma lúdica e interativa, cenário facilitado pela ampla acessibilidade estudantes ao uso de *smartphones* com o Android.

Por fim, destaca-se a liberdade de atualização e modificação das informações na base de dados do *Sem!o*, a qual pode ser realizada em "lote" (vários registros de uma vez). Esta característica amplia as possibilidades de utilização, tornando-o flexível e expansível e aplicável em qualquer tema ou área do conhecimento para o ensino-aprendizagem educacional. Trabalhos futuros consistem finalizar o desenvolvimento dos casos de uso na versão web, testá-los, validá-los e incluir melhorias nas funcionalidades do *app Sem!o*.

## REFERÊNCIAS


[1] Lee U., Han K., Cho H., Chung K., Hong H., Lee S., Noh Y., Park S., Carroll J. M.. Intelligent positive computing with mobile, wearable, and IoT devices: Literature review and research directions. Ad Hoc Networks, Volume 83, 2019, 8-24, ISSN 1570-8705, 2019.

[2] Urtiga, K. S.; Louzada, L. A. Telemedicina: Uma visão geral da arte. Congresso Brasileiro de Informática Médica, 2004.

[3] Costello E, Corcoran, M, Barnett, J.S, Birkmeier, M. Information and Communication Technology To Facilitate Learning for Health Professions. Online Learning, v.18, n.4, p.1–17, 2014.

[4] Martin Florence, Ertzberger Jeffrey, Here and now mobile learning: An experimental study on the use of mobile technology, Computers & Education, Volume 68, 76-85, ISSN 0360-1315,2013.

[5] Maciel, J. N. Protótipo de conferência multimídia e transmissão de dados de experimentos médicos em tempo real pela Web. (Monografia). Universidade Estadual do Oeste do Paraná. 2005.

[6] Jin, J.; Bridges, S. M. Educational technologies in problem-based learning in health sciences education: A systematic review. Journal of Medical Internet Research, v. 16, n. 12, p. 1–13, 2014.

[7] Mccoy, L. et al., Developing Technology-Enhanced Active Learning for Medical Education: Challenges, Solutions, and Future Directions. The Journal of the American Osteopathic Association J Am Osteopath Assoc, v. 115115, n. 44, p. 202–211, 2015.

[8] Free, C. et al., The Effectiveness of Mobile-Health Technologies to Improve Health Care Service Delivery Processes: A Systematic Review and Meta-Analysis. PLoS Medicine, v. 10, n. 1, 2013.

[9] Sheppard, Sheppard Software: we make learn fun. Disponível em <http://www.sheppardsoftware.com/web_games_vocab_med .htm >. Acesso set. 2017.

[10] Sporcle. Can you pick the correct medical term starting with 'R'. Sporcle Inc. Disponível em <https://www.sporcle.com /games/Bellelady/medical-terms-letter-r>. Acesso set. 2017.

[11] Mcgrath, John. Medical Terminology Quiz. Disponível em < https://play.google.com/store/*apps*/details? id=*app*inventor.ai_dot_dox.TheMTQuiz>.>. Acesso set. 2017.

[12] Mcgrath, John. Learn Medical Terminology. Disponível em <https://play.google.com/store/*apps*/details? id=*app*inventor.ai_dot_dox.InterActiveMT>. Acesso set. 2017.

[13] MGS. Terminologia médica (Free). Medical Group Software. Disponível em < https://play.google.com/store/*apps*/details? id=com.medicalgroupsoft.medical.directorymedtermsmultilang.free>. Acesso set. 2017.

[14] Ufostudio, Dicionário da Saúde Offline. Disponível em < https://play.google.com/store/*apps*/details?id=com.ufo.disease>. Acesso set. 2017.

[15] Lasemi. Aplicativo: SemioQuiz. Disponível em <https://play.google.com/store/*apps*/details? id=com.LASEMI.SemioQuiz&hl=pt_BR>. Liga LASEMI-UFBA. Acesso set. 2018. 2018.

[16] Kahoot. Kahoot!. Disponível em <https://kahoot.com/>. Acesso set. 2019.

[17] Alves, W. P. Projetos de Sistemas Web. Conceitos, Estruturas, Criação de Banco de Dados e Ferramentas de Desenvolvimento. Editora Érica. 1ª ed, 2015. ISBN-13: 978-8536510859.

[18] Mendoza A. D., Wolfgram E., Oliveira S. P., Maciel, J. N., Desenvolvimento do Protótipo de um Aplicativo para o Estudo da Semiologia Médica. Resumo - Anais do XVI Congresso Brasileiro de Informática em Saúde – CBIS 2018. Fortaleza-CE. 2018.

[19] Wolfgram E., Mendoza A. D., Oliveira S. P., Maciel, J. N., Definição de Requisitos e Construção do Banco de Dados de um Protótipo Educacional de Aplicativo Móvel para o Estudo da Semiologia Médica. Resumo - Anais do XVI Congresso Brasileiro de Informática em Saúde – CBIS 2018. Fortaleza-CE. 2018.

[20] Powers D. PHP: A Quick Reference. In: PHP 7 Solutions. Apress, Berkeley, CA. 2019.

[21] Silberchatz A.; Korth H. F.; Sudarshan S. Sistema de Banco de Dados. Editora Campus. 6ª Ed. ISBN 9788535245356. 2012.

[22] Fielding, R. T.. Architectural Styles and the Design of Network-Based Software Architectures. Ph.D. Dissertation. University of California, Irvine. 2000.

[23] Halili, F.; Ramadani, E. Web Services: A Comparison of Soap and Rest Services. Modern *App*lied Science; Vol. 12, No. 3; 2018. ISSN 1913-1844.

[24] Deitel P., Deitel H., Deitel A. An droid how to Program. Prentice Hall Press, Upper Saddle River, NJ, USA. 2013.

[25] DiMarzio J. F.,Beginning Android® Programming with Android Studio. 4ed., John Wiley & Sons, ISBN:9781119419334, 2016.

[26] Retrofit. A type-safe HTTP client for Android and Java. Disponível em <https://square.github.io/retrofit>. Acesso: 17 jul. 2019.

[27] Pressman R. S. Engenharia de Software - Uma Abordagem Profissional. Editora Bookman. 7ª Ed. ISBN: 9788563308337. 2011.

[28] Wolfgram, E. ; Mendoza, A. D. ; Maciel, J. N. ; Oliveira, S. P. . Protótipo Computacional de Apoio à Educação Médica: Análise de Requisitos e Projeto do Banco de Dados. Anais do VII EICTI-UNILA, 2018.