Relatório de viabilidade do Let's Fun!

1. Introdução

O *Let's Fun!* deverá ser um aplicativo *mobile* e um site *web*, onde pessoas possam marcar eventos e cotar preços em estabelecimentos de entretenimento e lazer. O nicho de negócio que se deseja explorar não é novo, mas a forma como aproveitá-lo é o grande diferencial. Assim, busca-se agilizar o processo de criação de eventos de entretenimento e lazer.

2. Objetivo do documento

Este documento tem como objetivo apresentar ao solicitante do *Let's Fun!* um estudo de viabilidade que determine se vale a pena realizar o investimento de concepção deste *app*. O estudo avaliará as quatro óticas de viabilidade:

- **Operacional**, que busca responder se o *software* realmente vale a pena ser desenvolvido e se a solução proposta é viável do ponto de vista operacional;
- **Técnica**, que estuda se há recursos (equipamentos e conhecimento) já existentes ou que possam ser adquiridos, que possibilitem atender às especificações do *software*;
- **Cronológica**, que visa determinar um prazo de concepção e entrega do *software*, verificando se ele é realizável ou impeditivo;
- **Econômica**, que informará os valores necessários para construir o *software*, assim como o tempo estimado para o retorno do investimento.

3. Contexto

O *Let's Fun!* deverá ser um *app mobile* e um *site web* onde as pessoas possam criar eventos de entretenimento e/ou lazer. Seu principal objetivo é facilitar que usuários agendem encontros — para rever amigos dos tempos de escola, rever amigos do tempo da faculdade, *happy hour*, encontros com familiares, etc em estabelecimentos parceiros.

Ele possibilita a criação de eventos, em que se pode escolher:

1. Data e hora;
2. Local;
3. Convidados;
4. Tipo do evento;
5. Serviços agregados.

Relativo ao *Local*, é possível obter informações de determinados estabelecimentos, que têm uma parceria com o *app*. Assim, é possível validar datas, capacidade de receber as pessoas, os serviços solicitados, cardápios etc. Relativo aos *Convidados*, é necessário armazenar algumas informações sobre eles para que assim as devidas comunicações sejam enviadas.

4. Solução proposta

A concepção do *app* poderá ser realizada utilizando, para a versão *web*, *back-end* em linguagem *Java* com *Spring Boot*, e *front-end* em *Angular*; ou *back-end* em *C#* com *.NET Core* e *front-end* em *React*. Para a versão *mobile* deverão ser utilizadas as linguagens *Swift* para dispositivos iOS e *Kotlin* para dispositivos Android. O banco de dados deverá ser *PortgreSql*. A infraestrutura será hospedada na nuvem *AWS* da Amazon, em servidores *linux*. Inicialmente, será

configurada apenas uma máquina com 16 GB de memória, um processador e 1 TB de armazenamento para o banco de dados.

As duas alternativas para a versão web serão avaliadas a seguir. Para a versão *mobile*, inevitavelmente o *app* deve ser concebido em ambas tecnologias para ser disponibilizado nas lojas virtuais, garantindo maior alcance de usuários. Por esse motivo, não é necessária uma avaliação comparativa para essa plataforma.

5. Viabilidade operacional

Esta análise de viabilidade visa apresentar o valor operacional agregado das propostas, ou seja, as características que cada uma delas possui. Os critérios do *PIECES — Performance*, *Informação*, *Economia*, *Controle*, *Eficiência* e *Serviços* — são utilizados para quantificar essas características.

A tabela a seguir mostra a aplicação do *PIECES* utilizando pesos (de 1 a 3) e notas (de 1 a 5), que auxiliam na avaliação. Os pesos representam a relevância de cada critério para o *Let's Fun!*, enquanto as notas são atribuídas com base no que se espera da proposta analisada. Para o cálculo final, é utilizada uma média ponderada.

	Peso	**Nota proposta Java**	**Nota proposta C#**
Performance	3	5	4
Informação	3	5	5
Economia	2	4	4
Controle	1	4	3
Eficiência	1	4	4
Serviços	1	4	4

No critério *Performance*, a proposta em *Java* obteve a nota final 15 em comparação à nota 12 da proposta em *C#*. Isso ocorre porque, ao utilizarmos *Java*, temos à disposição tecnologias mais maduras para otimização de cache, o que melhora significativamente o tempo de resposta. Em *Informação*, ambas as propostas receberam a nota máxima, 15, pois conseguem fornecer adequadamente as informações desejadas.

Em *Economia*, novamente houve empate, já que ambas possuem custos semelhantes para disponibilização da versão *web*, incluindo hospedagem, licenças etc.

Em *Controle*, a proposta em *Java* obteve uma nova um pouco maior, pois possibilita o uso de um *framework* muito famoso e robusto para seguranças de aplicações *web*, o *Spring Security*.

Em *Eficiência* houve novamente empate, pois ambas receberam nota 4, já que oferecem meios semelhantes para automatizar processos de trabalho.

Por fim, em *Serviços*, mais um empate, uma vez que ambas entregam serviços com a mesma qualidade.

Ao final, as notas de cada uma são:

- **Java**: (15 + 15 + 8 + 4 + 4 + 4) / 11 = 4,55;
- **C#**: (12 + 15 + 8 + 3 + 4 + 4) / 11 = 4,18.

Nota-se que a proposta em *Java* tem uma leve vantagem na ótica *Operacional*. Isso se deve ao mercado de desenvolvimento adotar amplamente essa tecnologia.

6. Viabilidade técnica

Essa viabilidade avalia a qualidade técnica das propostas, além de verificar se as tecnologias apresentadas são as mais adequadas no que diz respeito a questões técnicas e de mercado. Ambas têm grande aceitação e oferecem uma ampla gama de recursos para a construção de *softwares*.

Contudo, *Java* obtém uma pequena vantagem por ser mais difundida no mercado. A quantidade de desenvolvedores que trabalham com essa linguagem é maior, o que, consequentemente, proporciona mais opções de suporte.

Dessa forma, vemos que, independentemente da escolha entre *Java* ou *C#*, tecnicamente ambas as propostas são sólidas e viáveis.

7. Viabilidade cronológica

Esta viabilidade avalia o tempo que cada proposta levará para ser concebida e determinar até que ponto isto pode impactar negativamente ao prazo desejado para a disponibilização do *software*.

Neste quesito, a proposta em *C#* é mais vantajosa por oferecer um ambiente de desenvolvimento totalmente integrado, ao contrário de *Java*, que é conhecido por ser um verdadeiro "quebra-cabeça" em termos de configuração. Assim, para o tamanho do *software* em questão, com *C#* estima-se um prazo de 1 ano e 1 mês, enquanto com *Java* seriam necessários 1 ano e 3 meses.

Porém, por se tratar de um *software* ainda não disponível no mercado, a pressa pela disponibilização não é necessária, pois não existe concorrência. Portanto, ambos os prazos são aceitáveis.

8. Viabilidade econômica

Esta viabilidade visa mensurar economicamente a relação entre custos e benefícios das propostas, para que seja possível identificar qual delas proporcionará mais benefícios em relação aos custos envolvidos. Inicialmente, será avaliada a proposta em *Java* e, em seguida, a proposta em *C#*.

8.1 Proposta Java

Com base no fato de que podemos contar com um desenvolvedor *Java full stack*, um desenvolvedor para *Android/Kotlin* e *Swift*, além de um líder de projeto — sendo todos responsáveis também pela documentação —, temos os seguintes gastos iniciais com a equipe de desenvolvimento:

	Salário(mensal)	1 ano
Desenvolvedor full stack pleno	R$ 5.500	R$ 66.000
Desenvolvedor Android/Swift pleno	R$ 6.000	R$ 72.000
Líder de projeto	R$ 8.000	R$ 96.000
Totais	R$ 19.500	R$ 234.000

Após o período de desenvolvimento, os seguintes custos operacionais são esperados:

	Valor	1 ano
Nuvem AWS	R$ 500	R$ 6.000
Marketing	R$ 3.000	R$ 36.000
Totais	R$ 3.500	R$ 42.000

Um intervalo de 2 anos é usado como referência, com uma taxa de desconto de 6%, que geralmente é similar à inflação.

- $VP_1 = 1 / (1 + 0.06)^1 \rightarrow VP_1 = 0,94$
- $VP_2 = 1 / (1 + 0.06)^2 \rightarrow VP_2 = 0,89$

	Ano 1	Ano 2	Ano 3
Custo de desenvolvimento	R$ 234.000		
Custo de operação e manutenção		R$ 42.000	R$ 42.000
Fator de desconto	1%	0,94%	0,89%
Custos corrigidos	R$ 234.000	R$ 39.480	R$ 37.380
Custos acumulados	R$ 234.000	R$ 273.480	R$ 310.860
-	-	-	-
Benefício do *software*	R$ 190.000	R$ 190.000	R$ 190.000
Fator de desconto	1%	0,94%	0,89%
Benefícios corrigidos	R$ 190.000	R$ 178.600	R$ 169.100
Benefícios acumulados	R$ 190.000	R$ 368.600	R$ 537.700
-	-	-	-
Resultados acumulados	-R$ 44.000	R$ 95.120	R$ 226.840

A tabela anterior mostra que, a partir do segundo ano, o *Let's Fun!* feito com *Java*, começa a cobrir os custos de desenvolvimento, caso seja feito nesta linguagem. O período exato é:

- **FA** = |Quantia começo do ano| / (Quantia fim do ano + |Quantia começo do ano|)

No caso: **FA** = 44000 / 44000 + 95120 → **FA** = 44000 / 139120 → **FA** = 0,32. Ou seja, com 2,32 anos, o que é aproximadamente 1 ano e 4 meses.

O *ROI* para esta proposta é:

- **ROI** = Valor atual líquido / Custos totais

No caso: **ROI** = 226840 / 310860 → **ROI** = 72,97%

8.2 Proposta C#

Considerando que podemos contar com um desenvolvedor *C# full stack*, um desenvolvedor para *Android* e *Swift*, além de um líder de projeto — sendo todos responsáveis também pela documentação —, temos os seguintes gastos iniciais com a equipe de desenvolvimento:

	Salário(mensal)	**1 ano**
Desenvolvedor full stack pleno	R$ 6.300	R$ 75.600
Desenvolvedor Android/Swift pleno	R$ 6.000	R$ 72.000
Líder de projeto	R$ 8.000	R$ 96.000
Totais	R$ 20.300	R$ 243.600

Após o período de desenvolvimento, os seguintes custos operacionais são esperados:

	Valor	**1 ano**
Nuvem AWS	R$ 500	R$ 6.000
Marketing	R$ 3.000	R$ 36.000
Totais	R$ 3.500	R$ 42.000

Um intervalo de 2 anos é usado como referência, com uma taxa de desconto de 6%, que geralmente é similar à inflação.

- $VP_1 = 1 / (1 + 0.06)^1 \rightarrow VP_1 = 0,94$
- $VP_2 = 1 / (1 + 0.06)^2 \rightarrow VP_2 = 0,89$

	Ano 1	**Ano 2**	**Ano 3**
Custo de desenvolvimento	R$ 243.600		
Custo de operação e manutenção		R$ 42.600	R$ 42.600
Fator de desconto	1%	0,94%	0,89%

Custos corrigidos	R$ 243.600	R$ 40.044	R$ 37.914
Custos acumulados	R$ 243.600	R$ 283.644	R$ 321.558
-	-	-	-
Benefício do *software*	R$ 190.000	R$ 190.000	R$ 190.000
Fator de desconto	1%	0,94%	0,89%
Benefícios corrigidos	R$ 190.000	R$ 178.600	R$ 169.100
Benefícios acumulados	R$ 190.000	R$ 368.600	R$ 537.700
-	-	-	-
Resultados acumulados	-R$ 53.600	R$ 84.956	R$ 216.142

A tabela anterior mostra que, a partir do segundo ano, o *Let's Fun!* desenvolvido em *C#* começa a cobrir seus custos de desenvolvimento. O período exato é:

- **FA** = |Quantia começo do ano| / (Quantia fim do ano + |Quantia começo do ano|)

No caso: **FA** = 53600 / 53600 + 84956 → **FA** = 53600 / 138556 → **FA** = 0,39. Ou seja, com 2,39 anos, o que é aproximadamente 1 ano e 5 meses.

O **ROI** para essa proposta é:

- **ROI** = Valor atual líquido / Custos totais

No caso: **ROI** = 216142/321558 → **ROI** = 67,21%

9. Resumo de propostas

A tabela que segue sumariza os resultados das duas propostas:

	Proposta Java	**Proposta C#**
Investimento de desenvolvimento	R$ 234.000	R$ 243.600
ROI	72,97%	67,21%
Payback em anos	2,32	2,39

Na tabela anterior, verifica-se que a proposta em *Java* é a melhor opção, pois apresenta um custo de desenvolvimento menor, um **ROI** mais alto e um retorno mais rápido.

10. Conclusão

Ao final da avaliação de cada viabilidade, a tabela da sequência apresenta os resultados obtidos:

Viabilidade	Proposta C#	Proposta Java
Operacional		X
Técnica	X	X
Cronológica	X	
Econômica		X

Nota-se que a proposta de construir o *Let's Fun!* em *Java* apresenta resultados superiores em relação à implementação em *C#*. Diante disso, recomenda-se a adoção dessa tecnologia para o desenvolvimento do projeto.