# Preâmbulo GFP App

*Aplicativo Gerenciador Financeiro Pessoal (GFP)*

## Descrição do software

Nossa empresa tem passado por algumas dificuldades financeiras e, em um esforço derradeiro, o CEO solicitou um app que ajudasse no controle financeiro da empresa. Ele deseja fazer um controle acirrado dos gastos por setor da empresa, de forma que possa cortar excessos ou transformá-los em investimentos.

Serão usuários dessa aplicação o CEO e o Analista financeiro que só poderão realizar operações se estiverem logados na aplicação.

Para usuários anônimos só será possível acessar as áreas classificadas como públicas, se uma área não estiver classificada como pública deve ser considerada privada, portanto só pode ser acessada se o CEO ou o analista financeiro estiver logado.

Nessa aplicação desejamos ter uma área de notícias. Nela um usuário logado pode publicar informações sobre as medidas de ajuste fiscal para toda a empresa. Cada notícia deve ter um título, uma mensagem, um autor e pode ter uma imagem (url). Cada nova notícia deve ter as propriedades: título, mensagem, autor e imagem (url para imagem, opcional). Deve ser possível também editar e deletar mensagens. Usuários anônimos podem apenas ler as mensagens.

Para a atividade fim do software precisamos de uma área de apresentação do resumo da conta da empresa (dashboard). Nela devem ser mostrados, o extrato detalhado do último mês, receitas e despesas, e o balanço, ou seja, o saldo atual. Além do extrato detalhado do mês haver uma projeção de gastos de *X* meses seguintes (selecionável). Essa projeção deve se basear apenas nos lançamentos realizados e deve mostrar uma previsão de saldo aproximado para cada mês.

Para realizar lançamentos precisamos de mais uma área na aplicação. Nela poderemos listar os últimos lançamentos realizados, cadastrar receitas ou despesas da empresa e remover lançamentos. Cada lançamento deve ter nome (string), descrição (string), valor (número positivo), receita (boleano, o valor *false* representa uma despesa), categoria (string), uma data de lançamento (number), o número de repetições da despesa(number) e a repetitividade (string: anual, mensal, semanal ou diaria).

A fim de permitir uma melhor classificação de gastos, a categoria do lançamento não pode ser uma string aleatória, ela deve ser carregada do servidor remoto. Para criar uma nova categoria precisamos de uma nova área para gerenciar categorias, onde seja possível listar as existentes, adicionar, editar ou remover uma categoria.

Portanto a aplicação deve possuir as seguintes áreas:

- Uma área de login
- Uma área onde possam ser publicadas notícias sobre o ajuste fiscal da empresa
- Uma área onde possamos visualizar um resumo da conta da empresa (saldo atual e previsão de receita)

- Uma área de gerenciamento de lançamentos, onde seja possível listar os últimos, criar e deletar lançamentos
- Uma área de administração de categorias, onde elas sejam listadas, adicionadas e removidas

São áreas públicas da aplicação:
- Área de notícias (home)
- Área de login

# API

Para realizar o login devemos realizar HTTP/POST da URI **/api/login**. O formato de dados deve ser um json com as propriedades *login e senha*. Duas contas de login se encontram disponíveis e não é preciso criar outras:
- Usuário **ceo**
  - Senha **123**
- Usuário **financeiro**
  - Senha **123**

```
{
 login: 'ceo',
 senha: 123
}
```

O login não autorizado deverá receber um erro **401 - Unauthorized**.

## Identificação

Todas os item da base de dados (notícias, lançamentos ou categorias) possuem uma id. Essa identificação é gerada pelo backend quando o item é criado, portanto o desenvolvedor não pode criar uma id aleatória na hora de criar um item.

## Notícias

| Descrição | Método HTTP | URI | Dado a enviar |
|---|---|---|---|
| Carregar notícias | GET | /api/noticias | - |
| Criar notícia | POST | | Objeto notícia (sem id) |
| Editar notícia | PUT | /api/noticias/{id} | Objeto notícia (com id) |
| Deletar notícia | DELETE | | - |

Propriedades de um objeto notícia:
- título (string)
- mensagem (string)
- autor (string)
- data (number)
- imagem (url, string) [opcional]

```
{
  id: 5,
  título: 'Queda da bolsa de Tókio',
  mensagem: 'Qual será a influência na qualidade do sushi no Brasil?',
  autor: 'ceo',
  data: 1492487642799, // em milisegundos, desde 01/01/1970
  imagem: 'http://am990formosa.com/wp-content/uploads/2016/11/000-bolsa.jpg'
}
```

## Lançamentos

| Descrição | Método HTTP | URI | Dado a enviar |
|---|---|---|---|
| Carregar lançamentos | GET | /api/lancamentos | - |
| Criar lançamentos | POST | | Objeto lançamento (sem id) |
| Editar lançamentos | PUT | /api/lancamentos/{id} | Objeto lançamento (com id) |
| Deletar lançamentos | DELETE | | - |

Propriedades de um objeto lançamento:

- nome (string)
- descrição (string)
- valor (número positivo)
- receita (boleano, o valor *false* representa uma despesa)
- categoria (string)
- data (number)
- repeticoes (number)
- repetitividade (string: anual, mensal, semanal, diaria)

As propriedades repeticoes e repetitividade permitem o reuso de um mesmo lançamento, por exemplo para representar uma conta parcelada. Caso o lançamento não se repita as propriedades podem ser omitidas.

```
{
  nome: 'Compra de roupas',
  descricao: 'Compra de roupas para o inverno',
  valor: 499.99,
  receita: false,
  categoria: 'Vestuário',
  data: 1495890074552,
```

```
  repeticoes: 3,
  repetitividade: 'mensal'
}
```

Categorias

| Descrição | Método HTTP | URI | Dado a enviar |
|---|---|---|---|
| Carregar categorias | GET | /api/categorias | - |
| Criar categorias | POST | | Objeto categoria (sem id) |
| Editar categorias | PUT | /api/categorias/{id} | Objeto categoria (com id) |
| Deletar categorias | DELETE | | - |

Propriedades de um objeto categoria:

- nome (string)
- descrição (string)

```
{
  nome: 'Vestuário',
  descricao: 'Roupas e acessórios'
}
```

## Sobre datas

A data pode ser criada usando *Date.now()*.
Para visualizar executar *new Date(1495890074552), onde 1495890074552 é a data armazenada.*
Se esse lançamento for mensal e se repita 3 vezes, ele deve se repetir em

- **[Sat May 27 2017 10:01:14 GMT-0300] (data inicial)**
- **[Sat Jun 27 2017 10:01:14 GMT-0300]**
- **[Sat Jul 27 2017 10:01:14 GMT-0300]**

## Rotas da aplicação

- /noticia (home)
- /login
- /conta (dashboard)
- /lancamento
- /categoria

# AT

***Tendo por base o Preâmbulo GFP App, desenvolver os cenários, os componentes, os serviços e o mock da API da aplicação.***

Não se esqueça que devemos ter componentes stateful e componentes stateless.

Você deverá usar o modelo de projeto para desenvolver sua aplicação, lembrando de criar seus componentes stateful na pasta *app/components* e os componentes stateless na pasta *app/common*. Após a criação do componente e seu módulo (em sua pasta específica) acesse o arquivo *app/components/components.js ou app/common/common.js* para registar o módulo criado.

Sua aplicação deve também possuir os seguintes serviços:

- LoginService (controla o estado de login e sua realização)
- NoticiaService (gerencia as notícias)
- LancamentoService (gerencia os lançamentos da conta da empresa)
- CategoriaService (gerencia as categorias)

Nesse trabalho você precisa também:

- Utilizar em pelo menos 2 áreas diferentes do app, estados filho do ui-router para edição de itens
- Usar $http em pelo menos 1 serviço
- Usar $resource em pelo menos 1 serviço
- Usar ng-messages em pelo menos 1 componente stateless
- Utilizar o padrão Emissor de Eventos para expor dados dos componentes stateless

Use esse json de dados para criar seu mock do app.

Leia com atenção todas as rubricas, alguma cobrança pode ter ficado implícita no texto. Se isso acontecer será explicitado na rubrica.

# Rubricas

| Nº | Competência | Subcompetência | Itens da Rubrica | D | ND | Feedback |
|---|---|---|---|---|---|---|
| 1 | Empregar o framework AngularJS em aplicativos Web simples | | O aluno criou uma aplicação sem erros que possui: dashboard, área de categorias, área de notícias e área de lançamentos; onde a área de notícias é a área inicial. | | | |
| | | | O aluno empregou pelo menos 3 diretivas diferentes do framework angularjs | | | |
| | | | O aluno criou pelo menos cinco serviços, um para cada cenário da aplicação e para o login. | | | |
| | | | O aluno usou ao menos uma vez, em um componente stateless, os recursos da biblioteca ng-messages | | | |
| 2 | Desenvolver aplicativos | | O aluno utilizou o modelo de projeto corretamente posicionando componentes stateless na pasta common e componentes | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | Web orientados a componentes | | stateful na pasta components. | | | |
| | | | O aluno usou ao menos duas vezes o padrão Emissor de Eventos para expor dados dos componentes stateless | | | |
| | | | O aluno criou um componente stateful para cada cenário da aplicação | | | |
| | | | O aluno criou pelo menos 3 componentes stateless | | | |
| 3 | Empregar o framework ui-router | | O aluno criou os cenários da aplicação associando as rotas à componentes | | | |
| | | | O aluno associou rotas inválidas para a área inicial | | | |
| | | | O aluno utilizou pelo menos dois estados filhos, em cenários diferentes, para realizar edição de itens | | | |
| | | | O aluno utilizou pelo menos uma vez o objeto $state para transitar manualmente entre estados da aplicação | | | |
| 4 | Empregar o framework ui-bootstrap | | O aplicativo entregue utiliza o sistema de grid para posicionar elementos na tela corretamente | | | |
| | | | O sistema de grid foi utilizado para que a aplicação fique corretamente responsiva | | | |
| | | | O aplicativo enviado utiliza diretivas ui-bootstrap em pelo menos duas oportunidades | | | |
| | | | O aluno utilizou uma barra de navegação no aplicativo | | | |
| 5 | Criar aplicativos Web com conexão a um backend RESTFUL | | O aluno implementou a API do aplicativo no ambiente backend-less provido no modelo do projeto | | | |
| | | | O aluno utilizou em ao menos um dos seus serviços o serviço $http para acessar a API rest | | | |
| | | | O aluno utilizou em ao menos um dos seus serviços o serviço $resource para acessar a API rest | | | |
| | | | O aluno implementou o erro de logins não autorizados corretamente | | | |