**Abordagem corrente**

Web  
Componente Front-End da aplicação que, não só acarreta a *user interface* do projeto (client), mas também páginas que permitem a configuração específica de alguns eventos (admin).   
  
Desta forma, alterações de configuração tais como:   
 - O intervalo de tempo em que corre o crawler (comunicação com o *dispatcher*),  
 - Critérios de prémios dos jogadores (comunicação com o *regex*),   
 - Critérios de atribuição de badgers a jogadores ou treinadores (comunicação com o *regex*)   
  
passam a poder ser feitas por um utilizador administrador.

Dispatcher  
Componente responsável por lançar eventos automáticos que o *crawler* irá consumir. Para além disso, é no *dispatcher* que se encontram as ações que determinam as configurações definidas pela componente *admin –* referidas na componente anterior e é nesta componente que é são escritos na base de dados os valores obtidos pelo crawler. O *dispatcher* está também encarregue por lançar eventos detetados na informação obtida pelos crawlers (como por exemplo, um jogador marcou um hat-trick, ou é a décima vitória consecutiva do clube XPTO). A deteção da maioria desses eventos é definida pela componente *regex*.

Regex  
Nesta componente encontra-se toda a lógica que determina as condições necessárias para que um evento seja despoletado.   
De forma a reduzir a quantidade de condições necessária para que se capture os inúmeros eventos pretendidos (tais como 3 golos marcados num jogo, 5º jogo consecutivo sem marcar golos, 34º jogo consecutivo sem perder, etc.) optou-se por atribuir a cada entidade (clube, jogador ou treinador) um texto codificado que identifica diferentes eventos, por exemplo, “G” representa um golo, “W” representa uma vitória e “L” representa uma derrota.   
Assim, através de um sistema de regular expressions que vai percorrer os enormes textos dos utilizadores em questão, obtêm-se as respostas pretendidas e consegue-se detetar a ocorrência dos eventos definidos.

Crawler  
Componente responsável por recolher dados automaticamente de outros sites e retorná-los ao *dispatcher*. A única informação que o *crawler* recebe são dados de configuração e estes vêm por parte do *dispatcher*.

//DB Api  
//Componente que trata de toda a comunicação com a base de dados. Esta componente recebe os //valores que o *dispatcher* recebe do crawler, os valores que o *dispatcher* recebe dos administradores, //envia (quer sejam os valores automáticos, quer os que vêm da componente web).

Nao vou ter tempo de fazer a DB API, e ainda o Regex, Dispatcher e Crawler, vao estar no mesmo "Quadrado" onde os dados fluem bilateralemente ( Ao contrario de 1 way como está acima).

**Logo vais ter que alterar a imagem para reflectir isto.**

Se achas que vais ter tempo de fazer qualquer coisa no admin front-end, we keep things as they are, se nao, apaga todas as mentions de tal feature.

DB  
Base de Dados MongoDB.  
(Aqui vou meter os JSONs todos)

Server  
Componente servidora principal. Contém todos os endpoints da aplicação (Web Api) e é responsável por todo o processamento de Back-End da aplicação, tais como: autenticação, autorização, mensagens, envio automático de emails e comunicação com a componente de armazenamento de dados.

**Web Crawler**

Foi desenvolvido em NodeJS uma API que percorre o site [*www.zerozero.pt*](http://www.zerozero.pt/) de forma a obter o máximo de dados de todos os jogadores, treinadores, clubes de futebol, resultados e classificações das primeiras divisões dos escalões sénior e últimos de formação do país.  
  
A API utiliza a biblioteca *Crawler-js* para ler os valores das páginas HTML do site acima referido, os valores são em seguida mapeados para os documentos (modelos) existentes na nossa base de dados em Mongo, através da biblioteca *Mongoose*.  
  
Para além de permitir assim uma sólida quantidade de informação dos utilizadores principais na plataforma independente de insumos dos mesmos, juntamente com o *dispatcher* - que determina os períodos de “fetch” de informação - é possível obter dados cuja necessidade de que estejam o mais atuais possível sejam constantemente atualizados em curtos períodos de tempo. Tais como resultados de jogos, estatística, entre outros.  
  
(Esta componente encontra-se em fase de teste)