

LIVRE BLANC

CONCEPTION

D'UN EXTRANET

(VOIRE D'UNE APPLICATION WEB)

Auteurs :
Jean-Guillaume Dujardin
Hugo Averty

V2 – Juillet 2014

Un projet Extranet a ceci de particulier que ce n'est ni une application interne (SI), ni un site web. Un Extranet doit être un vecteur d'image tout en proposant des interfaces efficaces pour les utilisateurs.

Ce document vous propose un tour des bonnes pratiques pour la conception de ce type de projet.



TheCodingMachine
TCM://

PREAMBULE

QU'Y A-T-IL DANS CE LIVRE BLANC ?

Nous proposons une approche très pragmatique pour présenter les grands principes ergonomiques à respecter – selon nous – lors de la conception d'un Extranet. Nous nous sommes efforcés de produire un document synthétique mais complet en matière de recensement des bonnes pratiques.

En bref, ce livre blanc a pour objectif de dépasser la discussion purement techno pour amener un regard plus large sur les bonnes questions à se poser et les erreurs à ne pas commettre. Ces recommandations font suite à nos expériences de développement d'applications en général, et d'Extranets en particulier, chez TheCodingMachine. Ce sont les pratiques que nous avons mises en place au fur et à mesure des tests, et de l'observation des solutions qui fonctionnent le mieux, que ce soit en termes de développement ou de confort pour l'utilisateur final.

À QUI S'ADRESSE CE LIVRE BLANC ?

Evidemment, à ceux qui vont lancer les travaux de développement ou de refonte de leur Extranet. Il vous permettra d'avoir un éclairage objectif sur les différents aspects à avoir en tête au moment de créer un Extranet, afin que vous fassiez le choix le plus approprié à votre projet et votre équipe.

Toute l'équipe de TheCodingMachine se tient également à votre disposition pour discuter du sujet, et vous aider à concevoir et implémenter vos nouveaux projets !

PARTIE 1 - PERIMETRE D'UN EXTRANET

UN EXTRANET, POUR QUOI FAIRE ?

Un Extranet est une extension du système d'information d'une société à l'usage de ses partenaires (clients, fournisseurs...). Il a pour objectif d'offrir un accès aux informations techniques ou de gestions nécessaires pour qu'ils coordonnent leurs activités. Les exemples sont nombreux : il peut s'agir de proposer un catalogue de produits ou de services, de proposer un accès à la gestion des stocks ou aux factures, de mettre en place un programme de fidélité, d'automatiser un processus de gestion avec plusieurs interlocuteurs etc.

Les Extranets reposent, dans la plupart des cas, sur des technologies web : l'avantage est qu'il n'y a aucun logiciel à installer ; l'accès se fait simplement par mot de passe, depuis n'importe où.

L'Extranet est donc simplement un type d'application web pensé pour un besoin « métier » spécifique pour des utilisateurs extérieurs à la société.

Les avantages liés à la mise en place d'un Extranet sont multiples : faciliter le partage des informations, gagner en productivité (éviter les doubles saisies par exemple) ou bien encore sécuriser les transactions.

EN QUOI LA CONCEPTION D'UN EXTRANET EST-ELLE (UN PEU) DIFFICILE ?

On dit « difficile » mais pas infaisable (puisque vous avez ce livre blanc entre les mains !). Certaines contraintes doivent pourtant impérativement être prises en compte lors de la réflexion sur la création d'un Extranet.

DES CONTRAINTES SUR L'ASPECT ET LA PRISE EN MAIN DE L'OUTIL

L'Extranet est l'interface entre une société et ses partenaires. Il s'agit à la fois d'un outil de travail mais aussi d'un vecteur d'image. Une attention doit donc être portée à la fois à l'ergonomie (outil de travail) et au design (image).

A ces deux contraintes, construire un outil à la fois beau et pratique, s'ajoute une troisième contrainte : il faut souvent concevoir et/ou tester cette application sans avoir recours aux utilisateurs finaux. La plupart du temps ces utilisateurs sont des clients ; il est donc difficile de les solliciter.

ET UNE AUTRE CONTRAINTE SUR LA FACILITE D'USAGE ET D'APPRENTISSAGE

Le temps qui peut être consacré par les partenaires à l'apprentissage de l'outil est forcément court. L'utilisateur doit donc pouvoir trouver facilement et sans trop se poser de questions tous les éléments nécessaires à son travail.

Pourtant certains processus sont complexes par nature (songez au domaine de l'assurance par exemple). Comment faire en sorte de bien gérer ces deux états qui semblent contradictoires (temps de formation vs. processus complexe) ?

Il faut réduire l'application à l'essentiel, mais seulement à première vue ! L'application doit donc proposer des fonctionnalités simples pour la plupart des utilisateurs et des fonctionnalités avancées qui ne s'affichent que pour les utilisateurs les plus chevronnés.

ET PUIS CE QU'IL FAUT ÉVITER !

NE PAS IMITER WINDOWS

L'Extranet est souvent une extension d'outils internes préexistants. Le danger est donc de se retrouver avec un copier-coller des outils de type Windows (par exemple une base Access). Or, ce qui est acceptable par des utilisateurs en interne (formation longue, obligation d'utiliser l'application) ne l'est pas forcément par des partenaires. D'autre part, l'ergonomie client-lourd n'est pas du tout adaptée au fonctionnement sous un navigateur Web.

NE PAS COPIER LE WEB

Une tendance peut être aussi de copier le fonctionnement d'un site web classique. Or ce fonctionnement est rarement adapté au domaine des applications métiers. Par exemple, lorsqu'un utilisateur est sur un site e-Commerce, procéder par étapes pour passer une commande est plutôt intéressant : il est habile d'engager le client dans le processus de vente en proposant plusieurs formulaires simples. En revanche, sur un Extranet, proposer une fonctionnalité découpée en plusieurs étapes est une pratique qui risque de faire perdre beaucoup de temps à vos utilisateurs, et de finir par les agacer !

PARTIE 2 - LES GRANDS PRINCIPES

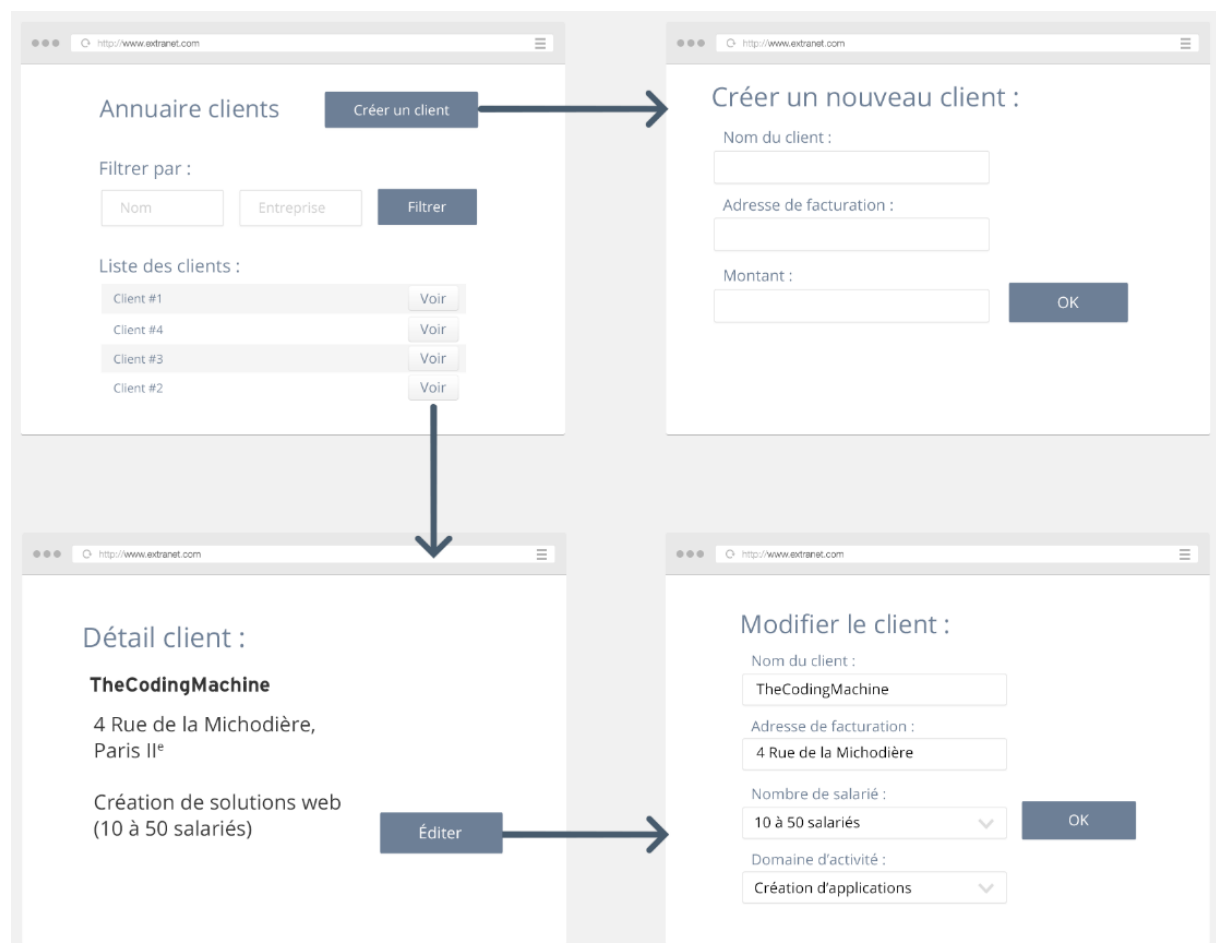
L'objectif de cette partie est de vous présenter les grands principes concernant les enchaînements des écrans d'une application. Ces principes peuvent sembler triviaux mais NE PAS RESPECTER CES PRINCIPES CONDUIT EN GÉNÉRAL À L'ÉCHEC DU PROJET ! (Vous ne pourrez pas dire que je ne vous ai pas prévenu, non ?)

L'ENCHAÎNEMENT LISTE / CONSULTATION / CREATION-EDITION

Le premier principe est de systématiser le fonctionnement de l'application sous la forme très classique d'un enchaînement liste / consultation / création- édition :

1. LISTE : une liste des objets métiers est présentée à l'utilisateur : il peut naviguer dans la liste ou faire une recherche par filtre.
2. CONSULTATION : un clic dans la liste permet à l'utilisateur de consulter le détail d'un item.
3. CREATION-EDITION : deux types de formulaires permettent au moment approprié de créer ou d'éditer un objet métier.

Les objets métiers peuvent donc systématiquement être présentés sous la forme de l'enchaînement suivant (exemple sur une liste de clients) :



La logique que nous venons de détailler peut sembler simple au premier abord, mais dans la pratique, appliquer ce système à l'ensemble des fonctionnalités de l'outil est plus compliqué à respecter qu'il n'y paraît !

L'enchaînement liste / consultation / création- édition présente trois avantages :

- Le premier avantage de ce principe est que, lorsqu'il est appliqué de manière systématique, il va permettre aux utilisateurs d'acquérir des automatismes concernant l'application et il va réduire considérablement le temps d'apprentissage (les autres fonctionnalités fonctionneront de la même manière).
- Le second avantage de ce principe est qu'il permet à l'utilisateur de savoir exactement ce qu'il est en train de faire. Il respecte en effet la règle « **1 page = 1 fonction = 1 objectif** », qui fait partie de la structure fondamentale d'une application web. Il s'agit de la logique selon laquelle chaque clic de l'utilisateur sur un lien va avoir pour conséquence de l'amener sur la page correspondant à ce lien. Cette logique est fondamentale pour que l'utilisateur sache à tout moment quelle action il est en train d'effectuer, et laquelle de ses requêtes est précisément traitée.
- Le troisième avantage de ce principe est de permettre aussi un développement et une mise en place des tests plus systématiques. Les développements sont donc plus robustes et moins coûteux.

DES OBJETS DANS LES OBJETS

L'utilisation de ce principe permet de construire une liste imbriquée dans l'interface de détail. Par exemple, l'utilisateur peut consulter le détail d'un client et avoir une liste qui présente les factures de ce client.



Note : lorsque l'on navigue d'un objet à un autre, l'utilisateur peut ne plus vraiment savoir où il en est. Dans ce cas, une bonne pratique consiste à indiquer le contexte dans lequel il se trouve. Cela peut par exemple prendre la forme d'un fil d'Ariane cliquable en haut de la page.

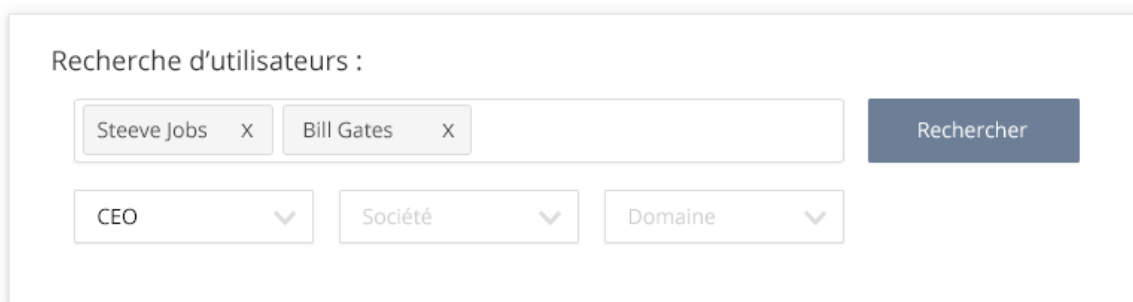
PARTIE 3 - LES GRANDES FONCTIONNALITES

L'objectif de cette troisième partie est de vous donner la plupart des bonnes pratiques qui fonctionnent sur les listes, les formulaires ou la gestion des erreurs.

ZOOM: LES FILTRES ET LES LISTES

GESTION DES FILTRES

La zone de filtres doit toujours se présenter sous la même forme. Nous aimons bien le principe adopté par tag-it pour mettre en valeur les mots clés de la recherche utilisateur :



La zone de recherche texte permet donc d'accueillir une saisie libre, ici le nom des utilisateurs, ou une sélection d'une zone de liste, ici le rôle utilisateur. Cet affichage permet de regrouper tous les filtres en un d'un point de vue de l'affichage et de transmettre l'information plus facilement à l'utilisateur.

Note : une bonne idée est aussi de ne présenter que les filtres les plus importants. C'est suffisant pour régler la plupart des recherches. Un lien « filtre avancé » permettra d'afficher tous les filtres possibles.

LES LISTES (EN GENERAL)

Pour afficher une liste, il sera nécessaire de choisir les bons éléments de détail à afficher. Ils permettent à l'utilisateur de voir en un coup d'œil quel enregistrement choisir. Si l'on reprend l'exemple précédent (une base client), si tous vos clients sont sur Paris, il vaut mieux afficher dans la liste les arrondissements plutôt que la ville.

Une bonne pratique consiste à mettre en place un tri sur l'en-tête des colonnes.

La pagination peut parfois être remplacée par un « infinite scroll » qui évite de charger une nouvelle page (évidemment, il ne faut pas que votre utilisateur scroll pendant des heures pour trouver l'enregistrement qu'il souhaite).

Note : Parfois gérer des actions sur plusieurs éléments à la fois peut simplifier la vie des utilisateurs (pensez à la fonction « archiver » de votre boîte e-mails). Dans ce cas, les éléments de la liste peuvent être sélectionnés (par une checkbox) et les boutons sont placés en bas de la liste.

*DANGER : Ce qu'il faut absolument éviter, c'est d'ajouter des actions de création/édition directement dans des listes. C'est en général une mauvaise pratique car ces éléments perdent l'utilisateur, qui risque de se demander par exemple s'il a mis à jour le bon élément. On revient toujours à la règle fondamentale **1 page = 1 fonction = 1 objectif** indiquée plus haut. De plus, cela complexifie le travail des développeurs donc cela augmente le coût et réduit l'évolutivité du projet.*

UNE LISTE N'EST PAS SUFFISANTE ?

Parfois les listes ne sont pas suffisamment explicites pour les utilisateurs. Dans ce cas, TheCodingMachine préconise d'utiliser une liste « améliorée » qui prend la forme suivante :



Un clic sur n'importe quel endroit de la ligne permet de faire apparaître une synthèse du détail d'un élément.

ZOOM: LES FORMULAIRES

STRUCTURE DES FORMULAIRES


La structure des formulaires peut être complexe (lorsqu'ils présentent de nombreux champs par exemple). Conserver une structure identique dans toute l'application est donc une bonne pratique. En général, un formulaire peut se structurer autour de 4 zones : une zone de contexte, une zone pour les erreurs (cf. gestion des erreurs), une zone de saisie et une zone pour les boutons d'action.


Ajouter un client

Vous ajoutez une facture sur le client TheCodingMachine

Merci de renseigner un code postal

Nouveau client :





AUTOUR DE L'INPUT (CHAMPS DE FORMULAIRE)

Les dernières études montrent que les formulaires les plus lisibles sont verticaux. Le libellé doit donc se situer au-dessus du champ associé (input en Anglais) :

Nouveau client :



Une autre bonne pratique est de glisser dans le champ un « placeholder » qui permet de donner plus d'informations à l'utilisateur sur la manière de remplir le champ, et qui disparaît dès que l'utilisateur commence à taper :

S'identifier :

Nom d'utilisateur

Mot de passe

LA BATAILLE DU SIMPLE OU DOUBLE COLONNE

Sur le web, il est déconseillé de gérer les formulaires en double colonne. Il vaut mieux fonctionner en plusieurs étapes. Pour un Extranet, c'est différent. Présenter un formulaire en deux colonnes peut simplifier la vie des utilisateurs en leur évitant de scroller pour visualiser les bonnes informations.

Même si le formulaire peut sembler complexe au premier abord, l'utilisateur va prendre des habitudes et il gagnera en efficacité à moyen termes.

UTILISER DES MODALES OU PAS ?

Parfois un formulaire n'est pas suffisant et utiliser une modale est indispensable. Une fenêtre modale est, dans une interface graphique, une fenêtre qui prend le contrôle total du clavier et de l'écran. Elle est en général associée à une question à laquelle il est impératif que l'utilisateur réponde avant de poursuivre, ou de modifier quoi que ce soit. Par exemple, un utilisateur peut avoir besoin de rechercher une valeur complexe à renseigner. Dans ce cas, l'utilisation d'une modale est indispensable. Une modale permet aussi de proposer des parties du formulaire pour les utilisateurs les plus avancés (sans que cela soit nécessaire aux autres).

Cependant, les modales doivent être utilisées avec parcimonie car elles interrompent la navigation et peuvent perturber les utilisateurs. Une bonne règle est donc de n'utiliser les modales que lorsque c'est indispensable (désolé, là, j'ai l'impression de ne pas beaucoup vous aider...).

GUIDER LES UTILISATEURS

Il convient de mettre en valeur les fonctionnalités logiques car tous les contrôles n'ont pas la même importance. Par exemple, sur une interface de création où il y aurait deux boutons : « Valider » et « Annuler », le bouton de validation aura plus d'importance que le bouton d'annulation car c'est l'action qui sera le plus fréquemment réalisée.



Votre nom d'utilisateur :

[Annuler](#) ou **Valider**

The image shows a user creation form. It has a label 'Votre nom d'utilisateur :', a text input field, and two buttons: 'Annuler' and 'Valider'. The 'Valider' button is highlighted with a dark blue background, indicating it is the primary action.

ZOOM: LA GESTION DES ERREURS

LES CHAMPS OBLIGATOIRES

Les labels des champs obligatoires seront en général indiqués par un *. C'est une convention que tout le monde commence à connaître.

LES CONTROLES DE SURFACE

Les contrôles de surface (format d'un e-mail valide, format ou longueur d'un champ etc.) sont dans la plupart des cas effectués en JavaScript et indiqués directement à l'utilisateur dans l'interface sous la forme suivante :

The diagram illustrates surface controls in a login form. It shows a transition from a generic form to a specific state with validation feedback.

Left Side (Generic Form):

- Label: S'identifier :
- Input field: Nom d'utilisateur
- Input field: Mot de passe
- Link: [J'ai oublié mon mot de passe](#)
- Button: Se connecter

Right Side (Specific State with Feedback):

- Label: S'identifier :
- Input field: Administrateur (with a green checkmark icon)
- Input field: ***** (with a red X icon)
- Message: Mot de passe invalide (in red text)
- Link: [J'ai oublié mon mot de passe](#)
- Button: Se connecter

An arrow points from the generic form to the specific state, indicating the transition from a valid input to an invalid password.

Note : une bonne pratique consiste à réaliser aussi ces tests côté serveur afin de garantir l'intégrité des données présentes en base.

LES CONTROLES METIERS

Les contrôles métiers ou de cohérence seront faits côté serveur. Ils sont effectués une fois que l'utilisateur a cliqué sur le bouton de validation du formulaire. Ces erreurs seront alors indiquées dans une zone de notification située au-dessus du formulaire :

CRÉER UN CLIENT

Attention : un client de type B ne peut pas avoir la propriété 5

test

✓

test@test.com

✓

Type B

✗

Propriété 5

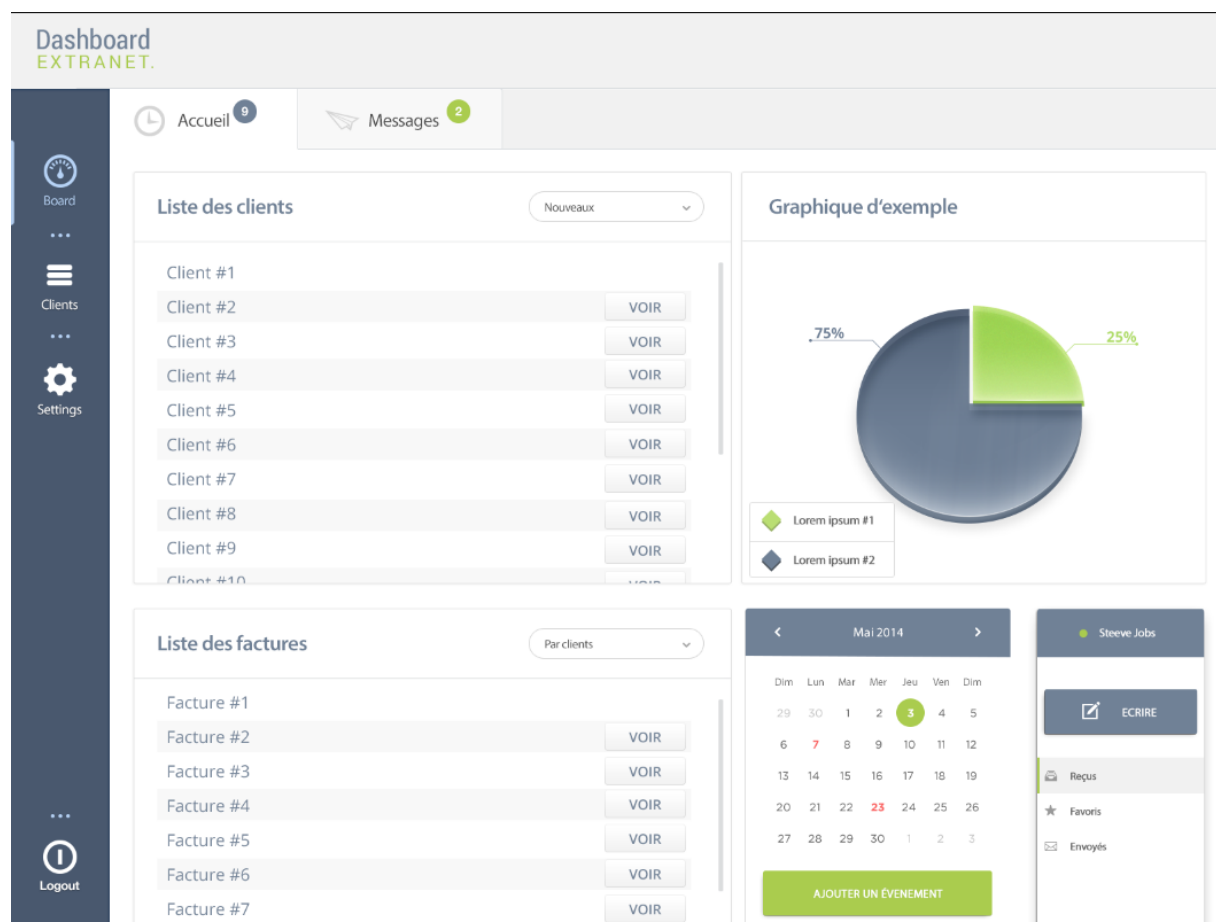
✗

Valider

ZOOM: LE DASHBOARD (OU PAGE D'ACCUEIL)

Il s'agit de la touche finale de votre application ! Paradoxalement, c'est un des éléments de conception que l'on fait en dernier. La première page peut être différente pour chaque famille d'utilisateur : l'espace de travail est personnalisé, l'application affiche les éléments dont le groupe d'utilisateurs aura besoin en priorité. Dans certains cas, on va montrer des informations statistiques (pour les gestionnaires par exemple), pour d'autres des tableaux présentant les dernières actions effectuées ou des notifications.

Dans tous les cas, cette première page est très importante. Elle permet de guider les utilisateurs dans les actions courantes qu'ils doivent effectuer et de réduire au minimum la recherche des informations les plus pertinentes.



CONCLUSION

Un Extranet est une application dont la conception est délicate mais nous espérons qu'avec toutes ces bonnes pratiques en tête vous pourrez vous en sortir !

Si vous vous posez d'autres questions, n'hésitez pas à consulter notre section Publications. Vous trouverez une mine d'informations sur : les Intranets, les performances, comment rédiger un cahier des charges etc.


N'hésitez pas non plus à nous dire si, selon vous, des éléments manquent, vous ont plu ou déplu. Bref, à venir discuter avec nous !

A PROPOS DE THECODINGMACHINE

TheCodingMachine accompagne ses clients sur des missions de conseil technologique et sur des projets de développement d'applications Web.

Nous sommes spécialisés dans le développement de sites Internet, d'extranets (évidemment), d'intranets, d'applications Web métiers en PHP et en JavaScript.

Fondée en 2005, TheCodingMachine a piloté plus de 200 projets. Nous travaillons aussi bien pour des grands comptes privés et publics, pour des PME-PMI que pour des startups. Nous avons investis dès notre création dans la R&D, ce qui nous permet par exemple d'être à la pointe des technologies web (PHP, Node.JS, AngularJS, streaming video etc.).



**Un projet, une idée,
TheCodingMachine peut
vous aider !**

TheCodingMachine
TCM://

www.thecodingmachine.com

Tél : 01 71 18 39 73

contact@thecodingmachine.com

4, rue de la Michodière – 75002 PARIS