

# Rapport - projet de groupe M1 - 2010

<b>1. Documentation commerciale.....</b>	<b>2</b>
• Pourquoi choisir notre solution ? .....	2
• Une architecture souple .....	2
• Une solution professionnelle .....	2
• Sécurité.....	2
• Audit.....	2
• Suppression "sûre" des données .....	2
• Internationalisation et localisation .....	2
• Une solution extensible.....	2
• Une solution qui suit la croissance de votre établissement .....	3
• Une focalisation sur la qualité .....	3
<b>2. Documentation utilisateur .....</b>	<b>3</b>
• Identification sur l'application (login) .....	3
• Création d'un compte .....	3
• Le calendrier .....	3
• Ajouter des événements personnels .....	4
• BackOffice .....	4
<b>3. Documentation administrateur .....</b>	<b>4</b>
• En tant qu'intervenant.....	4
• En tant qu'administrateur .....	4
• En tant que campus manager .....	4
<b>4. Guide d'installation de la solution .....</b>	<b>5</b>
• Pré-requis pour l'exécution de l'application.....	5
• Pré-requis pour le développement de l'application .....	5
• Préparation de la base de données.....	5
• Configuration .....	5
• Déploiement de l'application .....	5
<b>5. Documentation technique .....</b>	<b>6</b>
• Description de l'architecture.....	6
• Base de données .....	6
• Couche d'accès aux données .....	6
• Couche de services .....	6
• Couche applicative .....	7
• Détails de l'architecture de l'application web .....	7
• Web services / API .....	7
• Interface pour les téléphones mobiles .....	7
• Composants utilisés .....	7
• Any+Time™ DatePicker/TimePicker AJAX Calendar Widget.....	8
• FullCalendar .....	8
• Sécurité .....	8
• Rôles des utilisateurs .....	8
• Processus d'inscription .....	9
• Démarrage de l'application : gestion des rôles et de l'administrateur ..	9
• Sécurité des mots de passe.....	9
• Logging des actions utilisateurs .....	9
<b>6. Analyse et modélisation du projet.....</b>	<b>9</b>
• Modélisation UML du projet .....	10

# Documentation commerciale

## Pourquoi choisir notre solution ?

### Une architecture souple

L'architecture que nous avons retenue est extrêmement modulaire, permettant une évolution rapide, que ce soit pour l'ajout de nouvelles fonctionnalités ou pour l'intégration d'autres services.

### Une solution professionnelle

La solution que nous avons développée répond à des problématiques d'entreprise très importantes, détaillées ci-dessous :

#### Sécurité

Nous nous appuyons sur le framework Microsoft .NET pour garantir une application extrêmement fiable, et qui remplit des besoins plus que nécessaires :

- Suivi des utilisateurs : qui est connecté ? depuis quand ?
- Filtrage des utilisateurs : possibilité de bloquer des utilisateurs si leur comportement est jugé anormal
- Gestion des rôles des utilisateurs : l'utilisateur voit, selon ses droits, uniquement les informations auquel il a accès

#### Audit

Il est possible de suivre les modifications des objets de la base de données, permettant ainsi de résoudre des problèmes liés à des utilisateurs potentiellement mals intentionnés.

#### Suppression "sûre" des données

Aucune donnée n'est supprimée *réellement* en base de données, permettant une récupération de données effacées par mégarde extrêmement rapide.

#### Internationalisation et localisation

L'application est entièrement internationalisée : pour ajouter une nouvelle langue, il suffit de fournir à l'application les chaînes de caractères traduites.

### Une solution extensible

L'application dispose de web services, permettant facilement de créer d'autres applications autour du calendrier (par exemple, un widget pour Windows Seven, un widget pour iGoogle...).

Il est aussi très facile de réutiliser l'authentification de l'application pour d'autres

applications de votre écosystème, et de fournir ainsi un Single Sign-On (SSO).

### **Une solution qui suit la croissance de votre établissement**

Notre solution peut très facilement être étendue sur plusieurs serveurs, permettant ainsi d'accueillir plus d'utilisateurs tout en ayant la même qualité de service.

### **Une focalisation sur la qualité**

Durant toutes les étapes du développement, et ceux depuis la planification, une démarche qualité a été entreprise dans le but de fournir une solution des plus adaptés aux utilisateurs finaux, notre priorité.

## **Documentation utilisateur**

### **Identification sur l'application (login)**

La première page qui s'affiche lors de la connexion à l'application est un page de login. C'est à dire que l'application vous demande de vous identifier, en entrant votre nom d'utilisateur et votre mot de passe. En cochant la case "Remember me?", l'application se souviendra de vous et vous n'aurez plus à rentrer ces identifiants lors de votre prochaine visite.

SCREEN

Vous pouvez vous déconnecter de l'application en cliquant sur "log off" SCREEN

### **Création d'un compte**

Si vous n'avez pas encore d'identifiants pour l'application, vous allez devoir créer un compte en cliquant sur le lien "Register". SCREEN.

Vous devrez alors rentrer vos informations personnelles. SCREEN

Après validation du formulaire, vous pourrez vous identifier sur l'application mais vous n'aurez encore accès à rien. Pour accéder aux fonctionnalités de l'application, vous devrez attendre qu'un administrateur vous attribue un ou plusieurs rôles (élève ou intervenant, par exemple). Vous aurez ensuite accès à toutes les fonctionnalités vous concernant. SCREEN

### **Le calendrier**

Le calendrier est la partie centrale de l'application. Il présente l'ensemble de votre planning. Les différents couleurs indiquent différents types d'événements. SCREEN

En haut du calendrier, un jeu de boutons vous permet de naviguer dans le calendrier et d'afficher votre planning par mois, semaine, ou jour. SCREEN

Vous pouvez également filtrer les événements affichés grâce à plusieurs cases à cocher. SCREEN

En cliquant sur un événement, vous pouvez avoir des détails sur celui-ci.

## **Ajouter des évènements personnels**

Les évènements personnels sont des évènements que vous allez pouvoir ajouter vous même au calendrier. Il ne concernent que vous, et sont donc visibles de vous seul. Pour en ajouter un nouveau, il suffit de cliquer sur "Create an event". SCREEN  
Il faut alors remplir les champs du formulaire, puis valider. SCREEN  
Votre évènement apparaît alors sur votre calendrier. SCREEN

## **BackOffice**

Si vous êtes un intervenant, campus manager ou administrateur, vous avez également accès au backoffice (interface d'administration). Pour découvrir les fonctionnalités qui vous sont proposées, referez vous a la documentation administrateur. SCREEN

## **Documentation administrateur**

En tant qu'administrateur, campus manager ou intervenant, vous avez accès au backoffice (interface d'administration). SCREEN  
Ce backoffice permet de gérer différentes choses selon vos droits.

### **En tant qu'intervenant**

En tant qu'intervenant, vous allez pouvoir spécifier vos disponibilités a l'aide du backoffice. L'administration pourra ainsi planifier vos interventions en fonction de vos disponibilités.  
Pour cela cliquez sur "Manage availabilities" dans le menu du backoffice. SCREEN  
Vous accédez alors a une interface simple permettant de lister, ajouter, mettre a jour, supprimer vos disponibilités. SCREEN.  
Il vous suffit de remplir la date et l'heure de début de disponibilité, ainsi que la date et l'heure de fin. SCREEN.

### **En tant qu'administrateur**

En tant qu'administrateur, vous pouvez voir les utilisateurs, les approuver, ainsi que leur attribuer une classe et un ou plusieurs roles. SCREEN

Vous avez également accès a la gestion des campus, des cursus et des périodes d'études, pour mettre en place les nouveau cursus mais aussi pour gérer des maintenant les cursus déjà commencés en les ajoutant a l'application. SCREEN

Vous pouvez bien entendu ajouter des événements pour ces différentes périodes d'études. SCREEN

Vous pouvez aussi ajouter des évènements au niveau d'une campus, ou même des événements mondiaux. SCREEN

### **En tant que campus manager**

En tant que campus manager, vous avez les mêmes droits qu'un administrateur pour la gestion des utilisateurs, mais vous pouvez seulement attribuer le role student ou

stakeholder.

Vous pouvez aussi ajouter des évènements, soit au niveau des périodes d'études se déroulant dans votre campus, soit au niveau de votre campus. SCREEN

JE SUBIS TROP!

## Guide d'installation de la solution

### Pré-requis pour l'exécution de l'application

- Un ordinateur
- Windows XP / Vista / Seven
- .NET Framework 4.0
- SQL Server 2008
- Microsoft IIS 7.5

### Pré-requis pour le développement de l'application

En plus des pré-requis pour l'exécution, il est nécessaire d'installer :

- Microsoft Visual Studio 2010
- Microsoft SQL Server Management Studio 2008 (recommandé, mais pas obligatoire)

### Préparation de la base de données

Il est conseillé d'utiliser Microsoft SQL Server Management Studio 2008 pour réaliser ces opérations.

1. Créer une nouvelle base de données, nommée "**GlobusDataPad**".
2. Exécuter le script SQL situé à **/DAL/GDP.edmx.sql** pour créer les tables.

### Configuration

Selon le cas de figure, il peut être nécessaire d'adapter la connectionString du web.config pour le faire pointer vers un autre emplacement, si la base de données n'est pas sur le même serveur, ou si elle ne porte pas le nom de "**GlobusDataPad**".

### Déploiement de l'application

Il existe deux possibilités :

- Pour les développeurs en priorité, il est possible d'exécuter le projet via Microsoft Visual Studio 2010 et son serveur web intégré. Pour ce faire, après avoir ouvert la solution, il faut faire un clic-droit sur le projet "WebApp", puis cliquer sur "Start new instance" de l'item "Debug". Le navigateur web configuré par défaut s'ouvrira sur la page d'accueil du projet.
- Pour un déploiement en production ou en pré-production, il faut configurer un nouveau site via la console "IIS Manager", et positionner la racine du site sur le dossier "WebApp".

# Documentation technique

Il a été décidé d'utiliser exclusivement des technologies Microsoft et en particulier le framework .NET en version 4.0.

## Description de l'architecture

Il a été choisi de réaliser le projet avec une architecture N-Tiers nous permettant ainsi une flexibilité maximale.

### Base de données

Le stockage des données est effectué dans une base de donnée relationnelle. Nous avons choisi le serveur de base de données de Microsoft, SQL Server 2008.

### Couche d'accès aux données

Une première couche d'**accès aux données** (et en l'occurrence à la base de données Microsoft SQL Server 2008), le Data Access Layer, réalisé avec Microsoft Entity Framework en version 4. Les principales responsabilités de cette couche sont :

- description du modèle de données
- validation des données : règles pour valider les propriétés requises, le format des données (validation d'une adresse email, par exemple)
- gestion de la concurrence : si l'objet a été modifié *entre temps* par un autre utilisateur, alors une exception notifie l'utilisateur
- internationalisation : le nom de chaque propriété est internationalisé
- service générique pour persister des données

Du point de vue du développeur, il s'agit d'un projet "Bibliothèque de classes" contenant :

- un fichier EDMX (les entités générées à partir du modèle de données)
- un script SQL pour générer la base de données à partir du fichier EDMX
- les classes partielles correspondant à la personnalisation (internationalisation et validation des propriétés)
- des classes permettant d'accéder de manière générique à la base de données, et des attributs supplémentaires permettant, entre autre, de valider des adresses email, et de fournir un nom de propriété internationalisé

### Couche de services

Une deuxième couche de **services**, qui contient les méthodes métiers, et qui a comme responsabilités :

- renseignement des champs d'audit
- vérification des droits d'accès ("l'utilisateur a t-il le droit de faire cette action ?")
- persistance des données en appelant la couche d'accès aux données

Du point de vue du développeur, il s'agit d'un projet "Bibliothèques de classes" contenant des interfaces et leurs implémentations. Ces services représentent des parties logiques du domaine de l'application.

## Couche applicative

Une troisième couche : l'**application web**. Celle ci contient un minimum de logique : la plupart des traitements sont réalisés dans la couche services. Cette application contient aussi des vues pour les navigateurs mobiles.

Nous avons décidé d'utiliser le framework ASP.NET MVC 2, le modèle étant celui de l'application (c'est à dire celui de la couche d'accès aux données), et les contrôleurs appelant les services de la couche "Services".

### Détails de l'architecture de l'application web

Notre application web suit les principes de l'architecture MVC et l'implémentation définie par ASP.NET MVC 2.

L'application a été découpée en plusieurs parties (*Area*) :

- A la racine : un dashboard simple et les formulaires dédiés à l'enregistrement et à la connexion à l'application.
- Le *Front Office*, à savoir l'application de calendrier
- Le *Back Office*, à savoir l'application permettant de créer les événements de calendrier, les classes, cursus et autres entités associées. Il permet aussi de gérer les utilisateurs. Cette partie n'est accessible qu'aux utilisateurs qui ont pour rôles Administrateur, Campus Manager ou Intervenant, l'affichage étant différent selon les rôles.

### Web services / API

Nous avons mis en place des web services basiques permettant de récupérer facilement les différents types d'évènements pour les afficher sur le calendrier. Le format d'échange est le JSON.

Note : il est facilement possible d'étendre ces web services et de les proposer librement, par exemple, aux utilisateurs (qui pourraient ainsi établir leurs propres services). Il est aussi facile de proposer d'autres formats d'échange comme le XML.

### Interface pour les téléphones mobiles

La même application permet d'afficher des vues spécialement adaptées aux téléphones mobiles. Nous utilisons pour cela notre propre implémentation du WebFormViewEngine (le système qui retourne la vue selon la requête). Si le navigateur de l'utilisateur est repéré dans notre base de donnée comme étant mobile, alors on retourne une vue adaptée pour mobile - si elle existe, sinon une vue classique. (Il n'est pas recommandé d'utiliser le back office sur un navigateur mobile.)

### Composants utilisés

Nous avons décidé d'utiliser des composants déjà existants pour présenter certains contenus, notamment en ce qui concerne l'affichage de calendriers. Tous ces composants ont pour base **jQuery**, une librairie JavaScript simple et performante, très utilisé et reconnue. Nous utilisons aussi cette librairie (inclue de base avec ASP.NET MVC 2) pour la validation des formulaires côté client.

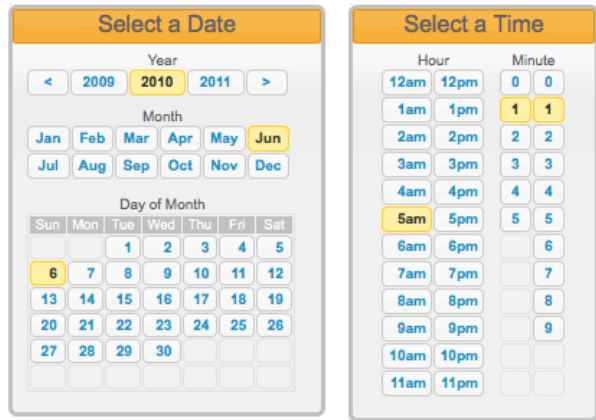
Nous avons veillé à ce que les composants soient stables et fonctionnent correctement

sur la majorité des navigateurs (Mozilla Firefox, Internet Explorer, Chrome, Opera, Safari...), et qu'ils aient une communauté active afin de permettre de résoudre plus rapidement des problèmes éventuels.

#### Any+Time™ DatePicker/TimePicker AJAX Calendar Widget

Nous avons fait le choix de ce composant permettant de choisir rapidement une date et une heure. Il nous a semblé très intuitif, présentant à la fois un calendrier permettant de se rendre compte du contexte de la date voulue (jour de la semaine...) et un sélecteur pratique de l'heure et des minutes.

Plus d'informations sur ce composant gratuit et open-source sur le site officiel : <http://www.ama3.com/anytime/>.



#### FullCalendar

Ce composant permet d'afficher des événements sur plusieurs affichages :

- par jour (*agenda*)
- par semaine (*weekly planner*)
- par mois



Ce composant présente beaucoup d'avantages, notamment une très bonne ergonomie (proche de Google Calendar), et de très bonnes performances (via une utilisation "juste" des web services que nous avons mis en place).

Plus d'informations sur ce composant gratuit et open-source sur le site officiel : <http://arshaw.com/fullcalendar/>.

## Sécurité

Nous avons décidé d'utiliser le framework Membership de ASP.NET, avec notre propre implémentation dans le but d'utiliser notre base de données. Cela nous permet d'intégrer dans notre modèle de données toutes les fonctionnalités de sécurité, et de permettre un accès facilité, tout en bénéficiant de la robustesse du framework fourni par Microsoft.

#### Rôles des utilisateurs

Nous utilisons aussi le framework Membership pour la gestion des rôles des utilisateurs. Nous avons défini quatre rôles qui peuvent être cumulables (un étudiant peut aussi être intervenant, par exemple) :

- Etudiant (*Student*)
- Intervenant (*Stakeholder*)
- Gestionnaire de campus (*Campus Manager*)
- Administrateur (*Administrator*)



Chaque rôle permet de "débloquer" une partie de l'affichage.

## Processus d'inscription

Nous avons défini une procédure stricte pour l'inscription des utilisateurs :

1. C'est à l'utilisateur de créer son compte via le formulaire "*Register*" à la racine de l'application
2. L'utilisateur est créé et verrouillé, dans l'attente de la validation d'un administrateur
3. Un administrateur approuve le compte dans le back office
4. L'utilisateur peut se connecter.

## Démarrage de l'application : gestion des rôles et de l'administrateur

Lorsque l'application est déployée (lorsque la DLL est publiée), l'application vérifie (via la méthode `Application_Start` de `Global.asax`) automatiquement que les rôles existent, et si ils n'existent pas les crée.

Il est aussi créé (si il n'existe pas déjà) un compte possédant le rôle d'administrateur et permettant de valider les nouveaux utilisateurs. Ce compte a pour nom d'utilisateur ("*Username*") "**admin**" et pour mot de passe "**adminadmin**". Il est important dans la mesure où c'est le seul utilisateur qui peut se connecter au back office au premier lancement de l'application.

## Sécurité des mots de passe

Les mots de passe sont hachés (*Hashed*) à l'aide d'un algorithme de hachage unidirectionnel et d'une clé de salage (*salt value*) générée de manière aléatoire au moment de leur stockage dans la base de données. Lorsqu'un mot de passe est validé, il est haché avec la clé de salage dans la base de données pour vérification. Il est impossible de récupérer ces mots de passe.

## Logging des actions utilisateurs

Chaque action de l'utilisateur est enregistrée, il est ainsi possible de connaître la date de dernière connexion, mais aussi de savoir si l'utilisateur est toujours actif (à chaque fois qu'il effectue une requête dans l'application). Il est aussi possible de désactiver automatiquement un compte qui aurait un trop grand nombre de tentatives de connexions avec un mauvais mot de passe.

Rappel : il est possible de connaître à tout instant la personne qui a créé et la dernière qui a modifié une entité.

## Analyse et modélisation du projet

## **Modélisation UML du projet**

A partir du cahier des charges, nous avons modélisé les entités comme des classes C# en utilisant les principes de la programmation orientée objet.

Il est ainsi très facile d'étendre le modèle de données et d'ajouter de nouvelles entités.

Note : à aucun moment nous n'avons réalisé de schéma relationnel pour une base de donnée relationnelle dans la mesure où tout est généré par Entity Framework. Nous nous concentrons uniquement sur la modélisation UML. Des scripts SQL de migration de la base de données sont éventuellement à prévoir si la base de donnée évolue.

