PROJET

Chacun doit présenter une fonctionnalité : 3 grosses fonctionnalités minimum.

Cahier des charges (MOA + MOE)

COMMéRé

Contexte : client, marché, besoin, existant

[présentation courte sur chaque point]

DEV-FLY est une filiale d'Air France créée en décembre 2012. Elle est spécialisée dans le marché low-cost du transport de passagers. Implantée à Paris, elle emploie 50 salariés. Sa flotte est constituée de 126 avions Airbus A321 de 170 places qui desservent 42 destinations.

Jusqu’à maintenant, Air France ne gérait pas les vols low-costs. C’est pour se faire une place sur ce marché porteur qu’elle a décidé de créer DEV-FLY. En effet, les vols low-costs devraient s’accroitre de 50% d’ici 2020, et Air France souhaite prendre part à cette croissance.

Jusqu’à présent, la filiale DEV-FLY avait recours au logiciel d’Air France, sur lequel une application spécifique avait été implémentée. Aujourd’hui, elle souhaite avoir sa propre application, afin d’avoir un outil personnalisé et évolutif, sur laquelle de nouvelles fonctionnalités pourraient facilement se greffer. Celle-ci devra couvrir :

* la possibilité de consulter et réserver des vols, pour 1 ou plusieurs passagers
* la gestion d’un espace client
* des fonctionnalités en back-office (visualisation du nombre de passagers et gestion de l’embarquement)

Le tout prendra la forme d’un site web. La gestion des bagages n’est pas prise en compte (chaque voyageur a droit à un bagage de 23kg en soute et d’un bagage à main de 10kg maximum, inclus dans le prix du billet – seule exception : les enfants de moins de 2 ans n’ont droit qu’au bagage à main).

Objectif(s) : fonctionnalités qui doivent répondre à des contraintes techniques (fiabilité + robustesse + performance)

[diagrammes de cas d'utilisation : 7 max, gérer la granularité. Accompagnés de scénarios (en faire 1 bien, ne pas tous les faire)

+ ensemble des IHM (maquettes...). (+ on peut faire 1 diag. d'activités).

Réponse de l'IHM en 2 secondes max]

Scénario de recherche d’un vol

|  |  |  |
| --- | --- | --- |
| Scénarios | IHM | Règles de gestion |
| La pers. se rend sur la page de recherche et saisit les critères de recherche : ville de départ et ville d’arrivée (à sélectionner dans une liste), date et heure d’aller, et éventuellement date et heure de retour. |  | Tous les champs doivent être renseignés et valides : les dates doivent être au format approprié, les villes doivent correspondre aux destinations prévues par la compagnie. Les villes de départ et d’arrivée doivent être différentes. Les dates doivent être incluses entre la date du jour et les 2 mois suivants. La date de retour (si renseignée) doit être ultérieure à la date de départ. |
| La compagnie renvoie une page de résultats. |  |  |
| L’utilisateur sélectionne un choix. |  | Le choix sélectionné doit correspondre à ce qui la compagnie propose. |
| La compagnie envoie un formulaire d’inscription de(s) passager(s) : civilité, nom, prénom, date de naissance. |  |  |
| L’utilisateur remplit le formulaire. |  | Tous les champs doivent être renseignés et valides : les dates doivent être au format approprié. La civilité doit être soit « Monsieur » soit « Madame ». Les noms et prénoms doivent être constitués de lettres. Au minimum un passager doit être majeur. |
| La compagnie envoie une fiche client à remplir pour inscription de(s) passager(s) : civilité, nom, prénom, adresse, code postal, ville, pays, mail, tél fixe, tél portable, login, mot de passe. |  |  |
| L’utilisateur remplit le formulaire. |  | Tous les champs doivent être renseignés et valides. La civilité doit être soit Monsieur » soit « Madame ». Les noms et prénoms doivent être constitués de lettres. Les numéros de téléphone doivent comporter des chiffres exclusivement, et peuvent commencer par « + ». L’adresse e-mail doit être au format correct, sans espaces ni caractères spéciaux (XXX@XX.xx). Le login doit être alphanumérique, et ne doit être déjà existant dans la base de données. Le mot de passe doit être une chaîne de caractères. |
| La compagnie envoie une fiche de synthèse + le formulaire de paiement. |  |  |
| Le client procède à la transaction. |  | Remarque : seul le paiement par carte bancaire est possible. |
| La compagnie génère et envoie au client un numéro de réservation. |  |  |

Remarque : si un champ dans un formulaire n’est pas conforme aux attentes, le formulaire est retourné avec un message explicatif en en-tête.

Un client aura différents choix possibles sur le site :

// diagramme de cas d’utilisation réservation

Pour réserver un billet, le processus sera le suivant :

1. Cas d’un nouveau client

// diagramme de séquence – cas où tout se passe bien

1. Cas d’un client déjà enregistré

Un client disposant déjà d’un compte ne remplira pas la fiche d’inscription.

On distingue 2 cas :

a) le client est déjà connecté à son compte lors de la recherche du billet :

// diagramme de séquence – client identifié

b) Le client n’est pas connecté à son compte lors de la recherche du billet :

// diagramme de séquence – client non identifié

Contraintes

* Seulement 3 personnes dédiées au projet.
* L’application doit avoir un temps de réponse de 2 secondes maximum avant d’afficher un résultat.
* La charte graphique de la société doit être respectée, une référence aux couleurs bleu / blanc / rouge doit être faite.

Moyens humains + techniques (environnement de développement : technologies, serveur, Apache...)

[Apache..., conception objet, MVC... Nombre de jours/homme. On pourrait faire un diagramme de déploiement.

Bien d'avoir chacun tous les rôles, au moins pour une fonctionnalité.]

Méthodes : cycle de développement, spécifications techniques (diagrammes)

[Spéc. techniques : diagramme de classe ou MCD pour décrire le modèle de données. Quels sont les objets + méthodes, attributs...

Diagramme de séquences pour une fonctionnalité.

Résultat : recette, présentation de l'application...

Méthode agile.

Principe de l'itératif (recommence) et de l'incrémental (rajoute une couche). Applicable sur de petits projets.

Pour chaque fonctionnalité : IHM + traitement métier + attaque DAO. Dans cet ordre ! Commencer par la fcté plus importante.

(La maquette se rapproche le plus possible de la réalité, plus ou moins selon à qui on s'adresse).

On ajoute à chaque fois une fcté à une fcté.

Au bout de 2 jours de dev : une première fcté à montrer. A isoler !! (cf versions)

Il faut TESTER l'application régulièrement. => tests à logger pour le jury.

Attention à la sécurité.