

Le Projet Tuteuré

Commentaires et Conseils

Table de matières

<i>Introduction</i>	3
Section 1. Le dossier de projet	3
Section 2. Conseils généraux	3
Section 3. Structure générale du dossier	4
1. Le rapport final.....	4
2. L'annexe technique	5
Section 4. La soutenance orale.....	6
Section 5. La démonstration.....	6
Section 6. Annexe 1. Cahier des charges	7
Section 7. Annexe 2. Dossier d'analyse des besoins	8
Section 8. Annexe 3. Dossier d'architecture.....	11
Section 9. Annexe 4. Dossier concepteur	13
Section 10. Annexe 5. Dossier de test.....	15
Section 11. Annexe 6. Dossier de maintenance	17
Section 12. Annexe 7. Dossier utilisateur	19

Introduction

La réalisation d'un projet tuteuré a de multiples objectifs: apprendre à travailler en groupe, se familiariser avec des outils et des langages, apprendre à définir, analyser, réaliser et tester un système informatique complexe.

Un autre des objectifs du projet tuteuré est aussi d'apprendre à présenter le travail réalisé. Ce dernier point est capital et mérite de s'y attarder. Il s'agit pour l'étudiant, d'apprendre à mettre en valeur son travail et pour les enseignants, de pouvoir évaluer le travail accompli.

La présentation du travail réalisé, comprend plusieurs aspects complémentaires :

- Un dossier,
- Une soutenance orale,
- Une démonstration ou une présentation de réalisation

Dans le présent document nous vous donnons quelques conseils visant à vous aider dans la préparation des différents documents.

Ce document est structuré de la façon suivante. La section 1 présente les objectifs du dossier de projet. La section 2 présente les qualités souhaitées pour le dossier. La section 3 propose une structure générale pour les différents composants du dossier. La section 4 présente quelques informations concernant la préparation de la soutenance orale. La section 5 présente quelques informations concernant la préparation de la démonstration et finalement les sections 6 au 12, montrent des exemples des différents documents techniques qui font partie du dossier de projet. Ces documents montrent le minimum d'information à fournir pour chaque activité du développement du projet.

Section 1. Le dossier de projet

En effet, la rédaction d'un dossier est un travail ardu et de lui dépendra le jugement porté sur l'ensemble du travail.

Le logiciel résout le problème posé. Ses qualités sont sa correction (résout-il effectivement le problème posé?), sa complétude (le fait-il dans tous les cas de figures possibles?), son efficacité, sa facilité d'utilisation, sa lisibilité (est-il compréhensible par d'autres?) et sa capacité à être maintenu et à évoluer. Certaines de ces qualités peuvent être testées (voire même automatiquement), mais c'est le dossier qui permet de comprendre la démarche ou la méthode suivie, la solution adoptée, les choix effectués, ... en un mot, le logiciel!. Mais le dossier de projet doit également permettre de prouver la correction du logiciel, d'avoir une évaluation chiffrée de ses caractéristiques techniques, l'évolution, la réutilisation et la maintenance (c'est à dire la correction de certaines erreurs qui ne manqueront pas de subsister) et enfin de l'utiliser correctement.

Avant de présenter la structure que nous proposons, pour le dossier de projet il nous semble convenable de donner quelques conseils de rédaction visant à faciliter la préparation du dossier.

Section 2. Conseils généraux

Un dossier de projet est un document technique mais il est destiné à être lu! Il est important que le dossier soit complet, mais aussi qu'il soit agréable à lire. D'autre part, chaque partie est destinée à un lecteur différent, il faut en tenir compte. D'une manière générale, il faut penser aux lecteurs et leur faciliter les différents types de lecture:

1. lecture exhaustive: ne pas se répéter mais faire des références précises (éviter les références en avant), éviter les romans fleuves (les gros pavés sont rarement appréciés) mais aussi le style télégraphique, faire des phrases courtes, n'écrire que sur le recto des feuilles en laissant une large marge tout autour, présenter un seul dossier dont toutes les pages s'ouvrent dans le même sens, etc.
2. lecture en diagonale: chaque partie doit être un ensemble bien structuré, agrémenté de schémas et d'exemples, il faut veiller à être cohérent entre les différentes parties.
3. lecture directe pour aller rechercher une information précise: il faut faciliter l'accès (table des matières, index, pages numérotées, figures numérotées, documents référencés, normes utilisées, etc.)

Voici une liste d'autres caractéristiques requises pour un dossier de projet :

1. Le dossier doit bien cerner la finalité du projet,
2. La présentation du travail effectué doit être synthétique
3. Le dossier doit être équilibré (en nombre de pages) entre le contexte de présentation du projet et le travail réalisé
4. Pensez à des schémas de synthèse pour faire passer les points importants
5. Tous les documents du dossier doivent être faits dans un français correct, dans un style clair et sans fautes d'orthographe. La présentation doit être soignée pour mettre en valeur le travail
6. Des éléments issues d'un document soumis à copyright (graphiques, images, etc.) ne peuvent être reproduites sans l'autorisation des ayants droits
7. L'organisation du rapport doit permettre de trouver rapidement les informations souhaitées

Section 3. *Structure générale du dossier*

Le dossier du projet (rendu fin mars) doit rendre compte de l'activité de l'étudiant pendant le projet. Il est constitué de deux parties principales :

- le rapport final : son objectif est de cibler le problème posé et de présenter les objectifs du projet. Ce n'est pas une juxtaposition des différents documents rendus au cours de l'année.
- l'annexe technique: les versions définitives des différents dossiers préparés au cours de l'année (cahier de charges, analyse de besoins, etc.)

Nous allons détailler ces différentes parties en donnant pour chacune son contenu (ou les informations que l'on souhaite y trouver), ses destinataires et quelques conseils spécifiques.

1. *Le rapport final*

Le rapport doit faire un bilan de la réalisation ou de l'étude ainsi que présenter l'apport du projet à la formation de l'étudiant. Par la suite, la structure générale du rapport final est montrée.

- Couverture
- Page de titre
 - Nom, prénom de chaque membre de l'équipe
 - Année scolaire
 - Nom de l'organisme pour lequel a été effectué le projet (dans notre cas IUT de Vélizy)
 - Titre du sujet
 - Nom du tuteur (s)
- Page de remerciements
- Table de matières : reprend tous les chapitres et leurs sub-divisions, avec leur numéro et leurs numéros de page.
- Résumé en anglais et en français
- Introduction
 - Présentation des objectifs du rapport
 - Présentation brève du projet :
 - objectifs
 - problème à résoudre
 - environnement de développement
 - Description de la structure du rapport
- Présentation de l'équipe de travail

- Nom et responsabilité de chaque membre de l'équipe dans le développement du projet.
- Présentation de la solution proposée et de la démarche utilisée
 - Méthode suivie
 - Les choix effectués
 - Qualités de la solution proposée : lisibilité, facilité d'utilisation, efficacité, etc.
 - La solution proposée résout le problème posé ?
- Analyse de la réalisation
 - Toutes les exigences ont été satisfaites ?
- Conclusions et bilan du projet
 - Résultats obtenus par rapport à ce qui était prévu, toutes les exigences ont été satisfaites ?
 - Les évolutions possibles.
 - Les acquis une fois le projet fini en termes de connaissances et de compétences.
 - Une organisation dans votre travail ?
 - Plus de rigueur dans le travail ?
 - Acquisition des méthodes ou des techniques de travail? Si oui, lesquelles?
 - Gestion du temps qui vous était imparti ?
 - Maîtrisiez-vous tous les outils qui étaient à votre disposition (logiciels et outils spécialisés...)
 - Les parties réutilisables.
 - Difficultés rencontrées (travail en équipe, apprentissage de nouvelles techniques et nouveaux outils, suivi de toutes les étapes.)
 - Améliorations apportées au delà du cahier de charges
- Bibliographie

Remarque :

- tous les documents faisant partie de l'annexe technique doivent être référencés de façon précise.
- toutes les images et graphiques doivent être référencés de façon précise.

2. L'annexe technique

Elle contient tous les détails techniques dont la présentation dans le rapport d'activité nuirait à la clarté de l'exposé. Les documents qui généralement font partie de l'annexe technique sont :

1. Cahier des Chargés
2. Dossier d'Analyse des Besoins
3. Dossier d'Architecture
4. Plan de Développement
5. Dossier Concepteur
6. Dossier de Tests
7. Dossier de Maintenance
8. Dossier Utilisateur
9. Le code source du logiciel développé en format électronique.

Cette liste peut être adaptée en fonction du type de projet. Des exemples de la structure souhaitée pour chacun de documents mentionnés précédemment, à l'exception du document de « Plan de

Développement », sont montrés dans les annexes. Un exemple d'un plan de développement type vous sera fourni ultérieurement.

Pour chacun de ces documents une page de garde et une table de matières doit être préparée.

Section 4. La soutenance orale

La soutenance du projet doit présenter de façon synthétique le travail qui a été réalisé afin de permettre d'apprécier les compétences que l'étudiant a mis en œuvre pour mener à bien le projet. Il ne s'agit pas de raconter le dossier du projet ni tous les détails techniques.

La présentation orale de projet a pour but d'apprendre à présenter de façon synthétique, à faire comprendre en un temps limité, à mettre en valeur un travail réalisé. Voici quelques conseils pour l'organisation de la soutenance :

La présentation : 30 minutes d'exposé (présentation et démonstration), 10 minutes de questions. Pour la présentation, une introduction en anglais sera exigée. Tous les membres de l'équipe doivent y participer. La présentation doit :

- mettre l'accent sur la finalité du projet
- cerner les principaux problèmes à résoudre et l'organisation du projet
- présenter rapidement les solutions envisageables (au niveau des principes et des outils utilisables) et argumenter le choix de la solution adoptée
- présenter de façon synthétique le travail effectué
- bilan : état d'avancement, difficultés rencontrées, apports du projet

Les documents et les supports utilisés pour la présentation :

- utiliser un vidéo projecteur, si possible, réaliser les diapositives avec un bon éditeur. Préparer de bonnes diapositives est très utile pour gérer le temps de l'exposé. Il est nécessaire d'avoir « répété » sa présentation pour s'assurer que le temps alloué est respecté. Dans le cas où le temps est dépassé, le jury peut demander d'arrêter la présentation
- ne pas oublier que les diapositives doivent être lisibles de loin, prévoir pour chacune :
 - un titre
 - un numéro
 - pas plus d'une dizaine de lignes
 - utiliser la typographie pour structurer le message que vous voulez présenter
 - chaque fois que c'est possible, utiliser des schémas, mais ne pas oublier qu'un schéma ne doit pas être trop chargé pour être lisible
- prévoir une diapositive de plan de l'exposé qui pourra être représentée pendant l'exposé pour situer son état d'avancement
- prévoir au minimum autant de diapositives que de points indiqués plus haut pour la présentation, sans oublier le bilan du projet

Section 5. La démonstration

L'objectif de la démonstration est de présenter le fonctionnement du logiciel qui a été développé dans le cadre du projet tuteuré. Cette démonstration doit être synthétique sans s'attarder dans tous les détails du fonctionnement, elle doit montrer seulement les fonctionnalités principales ainsi que le traitement des situations anormales. La démonstration doit être préparée.

Section 6. Annexe 1. Cahier des charges

<Nom du projet>	Version: <X.X>
Document : cahier de charges	Date: <jj/mm/aa>
Responsable de la rédaction :	

Cahier des charges

La rédaction du cahier des charges est la **première étape** de l'expression du besoin. Il s'agit d'un document rédigé au début du projet qui décrit les attentes des clients. Le cahier de charges peut être mis à jour en fonction des avancements dans le développement.

Le cahier des charges a pour but d'établir une description globale, à partir de la spécification des objectifs, des **fonctions qui doivent être fournies par le nouveau produit** ou des **extensions d'un produit existant**. Un tel document est indispensable : il permet à toutes les personnes impliquées dans le projet, de partager une vision claire du travail à réaliser. Ainsi, chaque acteur a connaissance des règles du jeu qui s'imposent à tous et il peut agir en conséquence.

En général le cahier des charges est **rédigé par le client** bien souvent en collaboration avec le responsable du développement. Il doit être **validé** pour s'assurer qu'il répond bien aux souhaits du client.

C'est un **document technique** sans considérations économiques. Il ne faut pas par exemple chercher à justifier le développement du produit en termes de rentabilité.

Il s'adresse au **client et aux développeurs** et sera à la base du contrat.

On reconnaît un bon cahier des charges aux **critères** suivants

1. il se place au bon niveau de **généralité**.
2. il **décrit sans ambiguïtés le problème posé**.
3. il définit des **critères de validation**.
4. il permet d'exprimer facilement un **changement** dans les besoins.
5. il doit être **concis**
6. il doit être **réaliste** : il ne s'agit pas d'imaginer un système idéal « qui fait tout » mais de rester réaliste par rapport aux besoins réels et à la faisabilité technique et financière

La structure, la précision et la longueur du cahier de charges dépendent de l'importance, de l'objet et du contexte du projet, il n'existe pas un format standard. Néanmoins, même si la présentation et l'ordre peuvent varier, les éléments décrits dans les paragraphes suivants doivent nécessairement y figurer :

1. Introduction

[Information générale sur le document, les objectifs du document, sa structure et les documents référencés]

2. Enoncé

*[La description **détaillée** du problème à résoudre, le contexte et les objectifs du projet, présentation de l'existant.*

Définition des objectifs qui doit atteindre la solution à développer]

3. Pré requis

[Connaissances requises, ressources matérielles et logicielles, compétences nécessaires]

4. Priorités

*[Les **priorités** éventuelles de développement, si elles ont été fixées avec l'accord du client]*

Section 7. Annexe 2. Dossier d'analyse des besoins

<Nom du projet>	Version: <X.X>
Document : analyse de besoins	Date: <jj/mm/aa>
Responsable de la rédaction :	

Analyse des Besoins

L'analyse de besoins permet d'identifier et de décrire précisément les besoins fonctionnels devant être satisfaits.

Les besoins doivent être décrits d'une manière

1. **précise** : les besoins doivent être bien délimités
2. **cohérente** : la description doit montrer que les besoins sont cohérents entre eux mêmes et avec l'environnement économique et technique
3. **complète** : la description doit tenir compte de tous les aspects.

Le dossier d'analyse de besoins doit être maintenable/flexible pour permettre la prise en compte des évolutions, dues à l'arrivée des nouvelles exigences ou à la modification des exigences existantes.

Le processus d'analyse et d'expression des besoins demande un effort de **communication** entre le client (utilisateur) et le fournisseur (unité de développement). La rédaction du dossier est confiée à l'**analyste** (à prendre ici comme un rôle) qui doit posséder les **qualités** suivantes :

1. capable d'**abstraction** et de **structuration** ; en effet les besoins de l'utilisateur sont exprimés souvent d'une façon confuse et contradictoire. L'analyste ne doit pas se perdre dans les détails.
2. capable de **connaître et anticiper le processus de développement** décrit dans le cycle de vie, par exemple il doit pouvoir prendre en compte des besoins liés à la maintenabilité du produit.

Le dossier d'analyse de besoins comprend :

1. Introduction

[Information générale sur le document, les objectifs du document, sa structure et les documents référencés]

2. Lecture du cahier de charges

2.1. Identifier les objets, les acteurs et les actions

2.2. Identification et description des exigences. La description d'exigences comporte entre autres :

- *le demandeur (source) de l'exigence.*
- *importance de l'exigence du point de vue de la "source". Les valeurs possibles (indicatives) de cette propriété sont:*
 - **Critique** : L'exigence est définitive. Le système ne peut pas exister sans elle.
 - **Mayeur** : L'exigence est très importante mais le système peut s'en passer pendant une courte période (initiale) provisoire.
 - **Mineur** : Il est souhaitable d'implémenter cette exigence, mais elle n'est pas indispensable pour le bon fonctionnement de l'application.
- *Version du produit dans laquelle l'exigence sera réalisée.]*

3. Analyse des interactions logiciel/environnement

[L'objectif de cette section est de présenter la spécification des interactions entre le logiciel et son environnement d'opération, relations avec les utilisateurs et avec d'autres systèmes]

3.1. Diagramme de cas d'utilisation

[Le diagramme de cas d'utilisation montre les fonctionnalités à implémenter par le système. Les fonctionnalités doivent correspondre aux exigences. Le diagramme est spécifié en la notation UML]

3.2. Spécification des cas d'utilisation

~~[La spécification de cas d'utilisation doit satisfaire le patron suivant, pour chaque cas d'utilisation :~~

~~**Nom du cas d'utilisation :**~~

~~**Objectif :** description de l'objectif du cas d'utilisation dans le contexte du système.~~

~~**Acteur Principal :** le nom et le rôle de l'acteur qui initie le cas d'utilisation par le biais d'un appel à une des actions du système.~~

~~**Autres acteurs participants :** d'autres acteurs qui sont en relation avec le cas d'utilisation pour accomplir un objectif bien défini.~~

~~**Point d'activation :** le nom de l'action sur le système qui déclenche le cas d'utilisation.~~

~~**Pré conditions :** c'est l'état qui doit être satisfait avant l'exécution du cas d'utilisation. Une pré condition est une condition présumée vrai.~~

~~**Post conditions :** c'est l'état qui doit être satisfait une fois l'exécution du cas d'utilisation finie. La post condition doit satisfaire les besoins des utilisateurs.~~

~~**Flot d'exécution basique :** description du scénario principal dans le cas d'utilisation. Le flot d'exécution basique n'inclut pas des conditions ou des branches. Une référence au diagramme de séquences qui spécifie le flot peut être ajoutée.~~


~~**Flots alternatifs :** description des extensions ou des variations du scénario principal. Une extension correspond à une branche ou à une condition dans le scénario principal. Un flot alternatif peut être représenté comme un autre cas d'utilisation.~~

~~Une référence aux diagrammes de séquences qui spécifient chaque flot alternatif peut être ajoutée.]~~

4. Étude de certains comportements

[Identification des différents scénarios et leur spécification en UML par le biais des diagrammes de séquences]

5. Spécification du comportement

 [Diagrammes d'états-transitions. Ce diagramme montre les changements d'états du système lequel est vu, dans la phase d'analyse de besoins, comme une boîte noire. Le diagramme doit être spécifié en UML]

6. Exigences non fonctionnelles

[Les conditions techniques ou non fonctionnelles, qui décrivent des contraintes liées aux plateformes, règles de sécurité, normes applicables, et ainsi de suite]

7. Exigences organisationnelles

[Les conditions de l'organisation, qui décrivent des contraintes liées à la réalisation des activités fournies par le système, aussi bien que leur intégration dans le nouveau système. Un diagramme d'activités peut être d'utilité pour la spécification de ces exigences.]

8. Prototypage de l'interface utilisateur

[Prototypes des différents moyens d'interaction entre le système et les utilisateurs (écrans, formulaires, rapports) et spécification de la carte de navigation. La carte de navigation permet de montrer la structure des dialogues entre le système et les utilisateurs]

9. Fiches de tests système



[Définition de tests d'acceptation,
Exemple :

Identification du test:		Version :		
Description du test :				
Ressources requises : (liste de ressources nécessaires, logiciels, matériels, données)				
Responsable : (nom du responsable de la conception du test)				
État initial	Cas testé	Acteur	Actions	Résultat attendu

]

10. Glossaire

[Liste de termes remarquables dans le contexte du problème et leur définition]

11. Conclusions

Section 8. Annexe 3. Dossier d'architecture

<Nom du projet>	Version: <X.X>
Document : dossier d'architecture	Date: <jj/mm/aa>
Responsable de la rédaction :	

Dossier d'architecture

Le dossier d'architecture permet d'établir la structure globale du système logiciel. Il montre différents modèles de l'architecture. Les modèles correspondent aux différentes vues, chaque vue représentant un ensemble d'éléments ainsi que leurs relations :

1. **Vue modulaire** : décrit ensemble d'éléments qui compose le système d'un point de vue statique et leurs relations.
2. **Vue composant/connecteur** : décrit les éléments du système qui sont présents au moment d'exécution
3. **Vue d'attribution** : décrit les relations entre le système logiciel et les éléments matériels ou humains présents dans son environnement

1. Introduction

[Information générale sur le document, ses objectifs, son organisation et les documents référencés. L'introduction rappelle également l'objectif et les caractéristiques du logiciel ainsi que le contexte du problème et les besoins fonctionnels et non fonctionnels]

2. Vue modulaire

*[Ce chapitre décrit l'ensemble d'éléments qui composent le système d'un point de vue **statique**. Il montre également la description des interconnexions entre les composants ainsi que les interfaces offertes par eux. La structure modulaire peut être spécifiée en UML par le biais de diagrammes de paquetages, diagrammes de sous-systèmes et diagrammes de classes.*

Ce chapitre doit aussi préciser le choix effectué par rapport au style d'architecture (en couches, etc.)

Quelques remarques :

- *Dans le cas d'un style en couches, le rôle de chaque couche est explicité et la présence de chaque couche est justifiée.*
- *Pour le diagramme de paquetages ou de sous-systèmes présenter le rôle de chaque sous-système ou paquetage, avec une courte description. Indiquer clairement si le composant est réutilisé tel quel, adapté ou développé. Présenter les classes importantes du paquetage.*
- *Pour les interfaces, description avec ses opérations.]*

3. Vue de composants/connecteurs

*[Structuration du système d'un point de vue **dynamique**. Le système est décrit comme un ensemble de composants avec un comportement bien défini au cours de son exécution. Pour ce type de structure un style architectural peut également être choisi (réparti, concurrent, données partagées, etc.). Cette vue peut être spécifiée à l'aide des diagrammes de sous-systèmes, des diagrammes de classes et des diagrammes de interaction].*

4. Vue d'attribution

[Description des associations entre les composants logiques et les composants physiques.

Les structures d'attribution permettent de spécifier :

- *Le déploiement de composants dans l'infrastructure matérielle. Les associations entre les composants et les nœuds d'exécution sont spécifiées. Chaque nœud et ses interconnexions*

sont décrits. La spécification, en UML, peut être faite à l'aide des diagrammes de déploiement.

- Les associations entre les composants et les éléments logiciel physiques (fichiers, tables, bibliothèques, etc. La spécification, en UML, peut être faite à l'aide des diagrammes de composants.]

5. Qualités de l'architecture

[Avantages et inconvénients de l'architecture par rapport aux critères de qualité exigés du système (portabilité, performance, tolérance aux pannes, etc. Limitations en vue d'une extension.]

6. Points ouverts

[Aspects de l'architecture qui méritent une amélioration ou une étude ultérieure.]

7. Glossaire

[Une description des termes utilisés dans les vues (modulaire, composant/connecteurs, attribution)

8. Conclusions



Section 9. Annexe 4. Dossier concepteur

<Nom du projet>	Version : <X.X>
Document : dossier concepteur	Date : <jj/mm/aa>
Responsable de la rédaction :	

Dossier concepteur

Le dossier concepteur est destiné à un éventuel programmeur qui veut comprendre la méthode de conception appliquée, le raisonnement suivi, les choix faits et les raisons de ces choix. Il doit y trouver tous les renseignements lui permettant de modifier, corriger, faire évoluer et réutiliser tout ou une partie du travail. Ce dossier montre le modèle de conception, lequel ajoute des concepts informatiques présents dans les outils, les langages de programmation ou les plateformes de développement.

1. Introduction

[Information générale sur le document, ses objectifs, son structure et les documents référencés]

2. Spécification détaillée de la structure du système

[Est un raffinement du diagramme de classes réalisé pour décrire l'architecture du système. Diagramme de classes de conception. Description des responsabilités de chaque classe. Spécification de chaque classe. La spécification doit montrer :

- 1. les attributs de chaque classe : nom et type de donné.*
- 2. les opérations de chaque classe : profil, pré et post condition, éventuellement un commentaire*

Le diagramme de classes doit également permettre l'identification des classes frontières, entités et contrôle.]

3. Comportement

[Optionnel. Cette section montre le comportement des différents composants du système. Spécifier des diagrammes d'interaction et des diagrammes d'états-transitions, pour les classes ayant un comportement dynamique.]

4. Spécification détaillée des interfaces utilisateur

[Description fonctionnelle de l'interface (comportement de tous les composants graphiques de l'interface). Spécification de la carte de navigation pour montrer la structure des dialogues entre le système et les utilisateurs]

4.1. Prototype de l'interface

[Les prototypes d'écran agissent en tant que modèle pour chaque composant dans le GUI.]

4.2. Définition de la charte graphique

[Couleurs, Logos, Typographie. Une charte graphique est l'ensemble des codes graphiques, colorés et formels créés pour la communication imprimée (ainsi que d'un site web). Cette charte a pour but de faciliter le quotidien de chacun en fournissant des modèles pour les documents les plus courants (papeterie, supports numériques, indications pour l'impression)....]

4.3. Spécification du site web

[Optionnel. Cette section est seulement applicable aux projets ayant un front-end web]

4.3.1. Définition de la charte graphique du site

[Couleurs, logos, typographie. Choix du fournisseur d'accès au site.]

4.4. Spécification de rapports et formulaires

[Donner le prototype des différents rapports et formulaires à être utilisés ou produit par le système.]

5. Modèle de réalisation

[Décrit la façon dont les éléments du modèle de conception sont implémentés par des éléments logiciels physiques de type : fichiers de code source, exécutables, tables, bibliothèques etc. La spécification, en UML, peut être faite à l'aide des diagrammes de composants.]

6. Politiques et stratégies

6.1. Mécanismes de stockage pour les données persistantes

[La quasi totalité des applications requièrent une base de données pour le stockage de l'information métier ou comme simple source d'informations. La forme de stockage de ces données (base de données, fichiers texte, fichiers tableur, images,...) Dans le cas de l'utilisation d'une base de données, spécifier le schéma entité/association ou le schéma relationnel]

Dans le cas de bases de données nominatives se préoccuper de la législation en vigueur pour la déclaration auprès de la CNIL.]

6.2. Politiques de sauvegarde des données persistantes

[Quelque soit la qualité des moyens de défense mis en œuvre (physique ou logiques) les données peuvent être altérées sciemment ou accidentellement.

Les données et les applications informatiques doivent être disponibles « à tout moment » lorsqu'on en a besoin, et doivent être conservées (sauvegardées) afin de pouvoir être récupérées (restauration) le moment voulu. Par exemple, il convient par conséquent de :

- 1. Définir une politique de sauvegarde ;*
 - a. Définir les périmètres à sauvegarder (services, matériels, sites, utilisateurs, ...)*
 - b. Définir le type de données sauvegardées (fichiers utilisateurs, fichiers serveurs, documents contractuels, emails, bases de données, ...)*
 - c. Fréquence/ périodicité de la sauvegarde, périodicité de la rotation des sauvegardes*
- 2. Définir des procédures de sauvegarde ;*
 - a. Sauvegarde complète*
 - b. Sauvegarde différentielle*
 - c. Sauvegarde mixte*
 - d. Sauvegarde incrémentale*
- 3. Définir des procédures de restauration ;]*

6.3. Politiques de sécurité

[La sécurité d'une application peut être compromise de multiples manières. Les applications web sont plus particulièrement vulnérables du fait de leur architecture distribuée et leur architecture n-tiers qui multiplient les composants autonomes représentant autant de maillons d'une chaîne de sécurité à rompre]

Pour réduire le risque de compromission des applications, il est nécessaire de se doter d'une politique de sécurité, exemples de ces politiques sont la gestion des privilèges, le cryptage des informations qui transitent sur le réseau, contrôle d'accès, etc.

[Vous devez définir l'ensemble des politiques de sécurité à mettre en œuvre].

7. Choix des outils à utiliser

[Langage de programmation, système de gestion de bases de données, bibliothèques. Choix d'un algorithme particulier ou patron de conception pour l'implémentation d'une fonctionnalité du système]

8. Conclusions et points ouverts

Annexe 5. Dossier de test.

<Nom du projet>	Version: <X.X>
Document : dossier tests	Date: <jj/mm/aa>
Responsable de la rédaction :	

Dossier de tests

Bien souvent destiné au même lecteur que le dossier concepteur, ce dossier doit néanmoins pouvoir être lu par n'importe qui. Il indique la procédure de test qui doit être effectuée pour garantir au maximum le bon fonctionnement du logiciel obtenu. Le dossier de test doit également montrer les résultats obtenus lors de l'application de la procédure de tests

Les tests sont effectués en plusieurs étapes :

1. Test unitaires et tests modulaires: ces tests peuvent être structurés selon deux critères complémentaires:

- par classe et par opération,
- par type de test: cas normaux, cas limites et cas d'erreurs.

Pour chaque cas, un tableau de test complet doit être présenté. Le dossier de test doit décrire tous les cas à tester.

Mais certains (faute de temps, d'espace...) peuvent ne pas avoir été exécutés.

2. Test d'intégration
3. Test d'acceptation
4. Test de système

Certains tests peuvent être exécutés automatiquement. Dans ce cas, donner les programmes de test développés.

1. Introduction

[Information générale sur le document, ses objectifs et les documents référencés]

2. Description de la procédure de test

[Description générale et justification de la procédure de test à être appliquée. Type de test, stratégie pour les tests unitaires (boîte blanche, boîte noire), ses objectifs et les documents référencés]

3. Description des informations à enregistrer pour les tests :

3.1. Campagne de test

[Définition du contexte des tests]

Exemple :

<i>Produit testé :</i>	
<i>Configuration logicielle :</i>	
<i>Configuration matérielle :</i>	
<i>Date de début :</i>	<i>Date de finalisation :</i>
<i>Tests à appliquer : (liste de références aux description des tests)</i>	
<i>Responsable : (nom du responsable de la campagne de test)</i>	

]

3.2. Tests

[Définition de chaque test

Exemple :

<i>Identification du test:</i>		<i>Version :</i>		
<i>Description du test :</i>				
<i>Ressources requises : (liste de ressources nécessaires, logiciels, matériels, données)</i>				
<i>Responsable : (nom du responsable de la conception du test)</i>				
<i>État initial</i>	<i>Cas testé</i>	<i>Acteur</i>	<i>Actions</i>	<i>Résultats attendus</i>

]

3.3. Résultats de test

[Définition de résultats de chaque test

Exemple :

<i>Référence du test appliqué:</i>
<i>Responsable : (nom du responsable de l'application du test)</i>
<i>Date d'application de test :</i>
<i>Résultat du test : (OK, KO, non fait, dérogé)</i>
<i>Occurrences des résultats :(éventuel, systématique)</i>

]

4. Conclusions

Section 10. Annexe 6. Dossier de maintenance

<Nom du projet>	Version: <X.X>
Document : dossier maintenance	Date: <jj/mm/aa>
Responsable de la rédaction :	

Dossier de maintenance

Le dossier de maintenance est un outil indispensable pour coordonner le processus de développement ainsi que pour contrôler les changements et les évolutions du logiciel. Ce dossier, comme le dossier concepteur, est destiné également à un éventuel programmeur qui veut comprendre le logiciel pour pouvoir le faire évoluer ou le corriger. Plus précisément, il faut y préciser les éléments suivants:

1. ce que fait effectivement le logiciel en fin de réalisation,
2. les cas non traités,
3. les erreurs ou incorrections détectées,
4. les évolutions prévues.

Le dossier de maintenance est destiné à évoluer en même temps que le logiciel. Chaque mise à jour, ajout ou correction du logiciel doit être répertorié dans le dossier de maintenance par un fiche de mise à jour décrivant la modification apportée au logiciel, la personne ayant effectué cette modification, la date et de nouveau en fonction de la modification ce que fait effectivement le logiciel, les cas non traités, les erreurs ou incorrections non corrigées, les évolutions prévues.

1. Introduction

[Information générale sur le document, ses objectifs et les documents référencés]

2. Règles pour la dénomination des éléments qui doivent être maintenus

[Chaque élément appartenant au système logiciel (spécifications, models, documentation, programmes) doit être identifié d'une façon unique. Les relations entre les éléments peuvent également être identifiés (un diagramme de composants UML peut être d'utilité)]

Définition des informations qui doivent être indiquées pour chacun des éléments/

Exemple :

<i>Nom du composant :</i>				
<i>Auteur du composant :</i>				
<i>Date de création du composant :</i>				
<i>Historique des modifications du composant :</i>				
<i>Numéro de version</i>	<i>Modifié par</i>	<i>Date de la modification</i>	<i>Modification effectuée</i>	<i>Raison de la modification</i>

3. État du système

[Description de ce qui fait le logiciel et des cas non traités]

4. Traitement des modifications

[Pour chaque anomalie détectée, une demande de modification peu être remplié]

4.1. Définition du formulaire pour la demande des modifications

[Exemple :

Nom du projet :	Référence :
Demandeur :	
Modification : (description de la modification demandée ou de l'anomalie à corriger)	
Responsable de l'analyse de la modification :	Date de l'analyse :
Composants affectés par la modification:	
Composant associés :	
Priorité :	Criticité (mineur, majeur, critique)
Estimation de l'effort : (combien d'heures seront nécessaires)	
Date d'envoi à l'équipe de programmation :	
Test de la modification : (références aux résultats de tests effectués sur la modification)	
Responsable de la réalisation de la modification :	
Date de la modification :	Date de retour à l'équipe de gestion de modifications
Commentaires :	

S'il s'agit d'une anomalie la référence du test échoué (KO) doit être également indiquée.]

5. Évolutions prévues

[Pour chaque composant la liste d'évolutions prévues]

6. Conclusions

Section 11. Annexe 7. Dossier utilisateur

Nom du projet>	Version: <X.X>
Document : dossier d'architecture	Date: <jj/mm/aa>
Responsable de la rédaction :	

Dossier utilisateur

Comme son nom l'indique, ce dossier est destiné à un utilisateur potentiel du logiciel, non nécessairement informaticien et encore moins programmeur; il est néanmoins inutile de lui apprendre à utiliser un ordinateur... Il faut donc donner à cet utilisateur tous les renseignements nécessaires et suffisants pour une bonne utilisation, ainsi que les renseignements concernant les problèmes pouvant surgir. En particulier, c'est ici qu'est décrite l'interface utilisateur, c'est à dire le déroulement du dialogue entre l'utilisateur et le logiciel.

1. Introduction

[Information générale sur le document, ses objectifs, son organisation et les documents référencés. L'introduction rappelle également l'objectif et les caractéristiques du logiciel ainsi que le contexte du problème et les besoins fonctionnels et non fonctionnels]

2. Présentation du produit

[Description du produit et du public à qui il est adressé]

3. Fonctionnalités

[Présentation des principales fonctionnalités du logiciel et description générale de l'interface avec l'utilisateur]

4. Mode d'utilisation

4.1. Procédure d'installation

[Décrire pas à pas la procédure à suivre pour la bonne installation du logiciel.]

4.2. Conditions d'opération

[Décrire les ressources nécessaires pour le bon fonctionnement du logiciel.]

4.3. Lancement

[Expliquer comment lancer le logiciel, les éventuels outils nécessaires et la structure des éventuels fichiers de données qui seront utilisés par le logiciel.]

4.4. Actions

[Décrire le traitement réalisé par les différentes commandes, les données, les résultats produits et les cas d'erreurs détectés. Il est nécessaire de préciser le type, la pré condition (souvent il s'agit d'un intervalle de valeur), le format d'entrée et le lieu de saisie des données sur l'écran. Un exemple d'exécution aide à la compréhension.]

La navigation des différentes fenêtres de l'interface utilisateur pourra être décrite à l'aide de captures d'écran.]

4.5. Erreurs

[Citer la liste des messages d'erreur. Une explication de la cause de l'erreur pourra également être donnée.]

5. Compatibilité avec d'autres produits

6. Conclusions