Plan du rapport du PFE d'ingénieur en informatique

proposé par Dr. Selma Belgacem
(les indications de l'encadrant restent toujours prioritaires)

- Il est fortement conseillé de rédiger le rapport en Latex.
- Utiliser "nous" au lieu de "je".
-Chaque figure ou tableau doit être interprété en utilisant sa référence (exemple : Dans la figure 3.1, nous présentons l'architecture du logiciel qui est composé de ...).
- Les références bibliographiques doivent être mentionnées au moment de leur utilisation dans le rapport (utiliser \cite{ref} en Latex).
- Faites attention au plagiat : pas de copier/coller, il faut reformuler avec vos propres mots.
- Les diagrammes mentionnés dans ce plan sont des diagrammes UML.

Remerciements

Table des matières

Liste des figures

Liste des tableaux

Introduction Générale

paragraphe 1 : décrire brièvement le domaine dans lequel votre projet se situe.
paragraphe 2 : décrire brièvement le sujet du stage.
paragraphe 3 : annoncer le plan du rapport.

1 Chapitre 1 : Présentation générale du projet

Introduction

Annoncer brièvement le contenu du chapitre.

1.1 Présentation de l'organisme d'accueil

1.2 Présentation du projet

1.2.1 Cadre général du projet

Décrire l'environnement du logiciel à développer, son contexte de fonctionnement, le domaine d'application...

1.2.2 Problématique

Les raisons qui ont mené à poser ce projet.

1.2.3 Solution proposée

L'idée principale de votre projet qui résoudra la problématique.

1.2.4 Objectifs

Les objectifs du projet doivent être compatibles avec les besoins fonctionnels et non fonctionnels.

1.3 Étude de l'existant

Système existant	Avantages	Inconvénients

→ L'utilité de ce tableau est de montrer tout au long du rapport, à travers les objectifs et dans le chapitre réalisation, que votre logiciel résout bien les inconvénients des systèmes existants.

1.4 Chronologie

- Décrire le calendrier de votre stage à travers par exemple le diagramme de Gantt.
- Vous pouvez mettre ce paragraphe dans le chapitre réalisation.

1.5 Processus de développement

Décrire le type de **cycle de vie du logiciel** appliqué (modèle en V, modèle en spiral, SCRUM...).

Conclusion

Résumez ce que vous avez présenté dans ce chapitre et faites le lien avec le prochain chapitre.

2 Chapitre 2 : Spécification des besoins

Introduction

2.1 Acteurs

Préciser et décrire les acteurs qui interagissent avec votre logiciel. Cette rubrique vous permet de préciser le type de l'utilisateur pour lequel votre logiciel est destiné, et par la suite, prévoir les besoins fonctionnels et non fonctionnels qu'il peut exprimer, et avoir une idée claire sur les principales fonctionnalités du logiciel qui doivent être facilement accessibles.

2.2 Besoins Fonctionnels

Les besoins exprimés par l'utilisateur qui sont équivalents aux objectifs du stage mais avec plus de détails. Ces besoins fonctionnels vont représenter plus tard les principales fonctionnalités de votre logiciel et par la suite les principaux cas d'utilisation (UML).

2.3 Besoins non Fonctionnels

Généralement, les besoins non fonctionnels correspondent aux indicateurs de qualité du logiciel (fiabilité, efficacité, ergonomie (facilité d'utilisation, IHM confortable...), performance (temps de réponse/temps réel...)...).

2.4 Diagrammes de cas d'utilisation

2.4.1 Diagramme général

- Donner un diagramme de cas d'utilisation général qui comporte par exemple trois cas d'utilisation : CU1, CU2 et CU3. Puis, détailler dans des paragraphes séparés chaque cas d'utilisation à travers son diagramme de cas d'utilisation.
- Il est possible d'utiliser un **diagramme d'activité** pour décrire l'enchainement général des fonctionnalités de votre logiciel.

2.4.2 Diagramme CU1 détaillé

Après avoir donné le diagramme CU1, choisir le cas d'utilisation le plus important et le décrire avec la **description textuelle**, ensuite, représenter le scénario nominal par un **diagramme de séquence "système"**.

2.4.3 Diagramme CU2 détaillé

2.4.4 Diagramme CU3 détaillé

Conclusion

3 Chapitre 3 : Conception

Introduction

3.1 Vue statique de l'application

3.1.1 Architecture logique du logiciel

Vous pouvez appliquer ici un **patron d'architecture** (Pipes and Filters, blackboard, reflection, clean...). Il est possible aussi d'utiliser les diagrammes UML (Diagramme de paquetage, diagramme de composants). Il ne faut pas nommer les technologies utilisées à ce niveau.

3.1.2 Diagrammes de classes

- Essayez de faire le lien entre l'architecture du système et le diagramme de classe. Exemple : si vous représentez l'architecture du logiciel avec un diagramme de paquetage (indépendamment du type du logiciel), reprendre chaque package et le représenter avec un diagramme de classes.
- Les diagrammes de classes peuvent contenir des **patrons de conception** (Singleton, Factory, Observer, Strategy...).
- Interprétez le diagramme de classes en décrivant les principaux attributs et les principales méthodes des principales classes.

3.1.3 Bases de données

Vous pouvez décrire le modèle de vos données en utilisant les diagrammes standards de description des bases de données comme le **diagramme entités-associations**.

3.2 Vue dynamique de l'application

3.2.1 Diagrammes de séquences

Un diagramme de séquences détaillé permet de décrire un scénario de communication entre les objets du logiciel afin de réaliser un cas d'utilisation (choisir les principaux cas d'utilisation réalisés par votre logiciel).

3.2.2 Diagrammes d'activités

Un diagramme d'activités permet de décrire un enchainement algorithmique interne au logiciel (choisir les principaux algorithmes dans votre logiciel).

3.2.3 Diagrammes états-transitions

Un diagramme d'états-transitions permet de décrire le processus de changement d'états d'un objet dans votre logiciel (choisir les objets les plus importants).

Conclusion

4 Chapitre 4 : Réalisation

Introduction

4.1 Technologies

Décrire brièvement les technologies utilisés pour développer votre logiciel (APIs, Frameworks, langages, environnements...).

4.2 Outils d'implémentation

Ici, vous allez décrire principalement le matériel utilisé pour développer le logiciel.

4.3 Architecture physique et évolution temporelle

- Mettre dans cette sous-section un **diagramme de déploiement** du logiciel qui décrit la superposition de l'architecture logique du logiciel (composition du logiciel) et son architecture physique (matériel). Ce diagramme permettra de décrire l'application des technologies utilisées dans le cadre du projet.
- possibilité d'ajouter des **diagrammes de temps** qui décrivent l'enchainement temporel précis avec une durée de traitement calculée pour les principales fonctionnalités du logiciel.

4.4 Interfaces de l'application

Dans ce paragraphe, vous mettez les imprimes écrans qui représentent les fonctionnalités principales de votre logiciel avec leur interprétation.

4.5 Résultats et statistiques

Si vous avez des résultats d'exécution hors les interfaces graphiques, ou si vous avez des statistiques d'utilisation de votre logiciel, vous pouvez les mettre et les décrire dans ce paragraphe.

4.6 Problèmes rencontrés

Dans ce paragraphe, vous pouvez décrire les difficultés rencontrées pendant le développement de votre logiciel et éventuellement comment vous avez pu les surmonter.

Conclusion

Conclusion Générale

paragraphe 1 : décrire brièvement le contexte d'application de votre logiciel dans le cadre de l'organisme d'accueil.

paragraphe 2 : résumer brièvement le travail réalisé.

paragraphe 3 : Donner quelques perspectives, qui sont réellement faisables, pour votre logiciel. Exemples : une possible amélioration du logiciel, des fonctionnalités qui peuvent être ajoutées au logiciel, d'autres cadres possibles d'application de votre logiciel.

Bibliographie

[numéro] Nom de l'auteur, "*Titre de l'ouvrage*", Journal/Institut/conférence/éditeur, date de publication.
→ Vous pouvez mettre les références des livres, des articles, des rapports d'anciens étudiants utilisés, des notes de cours, des documents fournis par l'organisme d'accueil.
→ Avec Latex, les items de la bibliographie ont un mode de génération automatique.

Netographie

[numéro] URL du site, consulté le ...

Annexe

L'annexe peut être considéré comme un ou plusieurs chapitres supplémentaires dans le rapport ou vous pouvez expliquer principalement les détails théoriques des méthodes ou modèles standards que vous avez appliqués (comme par exemple Blockchain, le Deep Learning...) avec éventuellement des exemples de clarification → Bref, le travail réalisé par d'autres spécialistes et pas par vous même, ce qu'on appelle aussi "État de l'art".
L'annexe permet d'alléger le contenu principal du rapport et en même temps permet d'ajouter des clarifications.