Ministère de l'enseignement supérieur de la recherche et de l'innovation

UFR Sciences et Technologies

Département d'Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de master

Mention : Informatique

Spécialité : Génie logiciel

Sujet :

Conception et mise en œuvre d'un système de vote électronique : application aux élections organisées à l'UASZ

Présenté par : M. Adama SEYE

Le 23/03/2019

Membres du jury :

- Pr. Salomon SAMBOU (**Président du jury**)
- Dr. Khadim DRAME (**Encadrant**)
-Dr Ibrahima DIOP (**Rapporteur**)
- Dr. Youssou FAYE (**Rapporteur**)
-Dr Khalifa GAYE (**Examinateur**)
-M. Alain Charles GOMIS (**DRH**)
-M. Alioune Badara DIENG(**CSP**)

Sous la direction de :

- Dr. Khadim DRAME

Sous la Supervision de :

- Pr Salomon SAMBOU

Année Universitaire : 2018 - 2019

RESUME

L'Université Assane SECK de Ziguinchor fonctionne grâce à des entités bien structurées en son sein. L'organisation hiérarchique de l'Université définit un ensemble de postes occupés par le personnel qui y travaille.

Certains de ces postes nécessitent une élection. Au sein de l'Université, un ensemble d'élections est organisé pour des groupes bien spécifiques tels que le personnel enseignant et de recherche (PER), le personnel Administratif, technique et de services (PATS) et les étudiants. Chaque élection est supervisée par un responsable de service (DRH, CSP, etc.). L'organisation des élections à l'UASZ est très pénible. En effet, elle prend beaucoup de temps sur sa mise en place (ouverture et fermeture des dépôts de candidats souvent rallongée). Le taux de participation est souvent faible. La note d'information n'est pas reçue par tous les concernés et enfin, des erreurs apparaissent sur le comptage des voix ce qui entrave la transparence du vote.

C'est ainsi que dans ce sujet de mémoire, nous comptons mettre en place un système de vote afin de répondre mieux aux exigences détaillées par le DRH, le Chef de Service Pédagogie, et même les étudiants. Après des enquêtes et des entretiens menés auprès des acteurs concernés, nous avons penché notre travail sur le cycle de développement SCRUM. Néanmoins, une spécification claire nous a permis de définir des modèles de conception grâce au langage de modélisation UML, le stockage des données est assuré par le SGBD MySQL. Pour l'implémentation, nous avons fait appel au langage Java qui nous a permis de disposer d'un prototype permettant de gérer une élection de son ouverture à sa fermeture en permettant l'envoi des notes d'information aux électeurs, les dépôts de candidatures, le vote, le suivi en temps réel des statistiques du scrutin, l'impression et l'envoi par mail des PV.

ABSTRACT

Assane SECK University of Ziguinchor is a full-fledged organization operating through well-structured entities within it. The hierarchical organization of the university defines a set of positions occupied by the staff who work there.

Some of these positions require an election. Within the university, we have a set of elections organized on very specific groups such as teaching and research staff (PER), administrative, technical and service staff (PATS) and students. A person (HRD, pedagogical officer, etc. supervises each election ...). The organization of these elections can be very delicate. Indeed, the traditional election (that is to say with the ballot box) is painful and can take a long time; the participation rate in an election is often low because the information note does not pass enough. There are also errors that can occur on counting votes. In addition, we have visible problems with the transparency of the vote.

Thus in this subject of memory, we intend to set up a voting system in order to meet as much as possible the detailed requirements by the HRD, the pedagogical officer and even the students. After surveys and interviews with stakeholders, we looked at the SCRUM development cycle. Nevertheless, a clear specification allowed us to define design models thanks to the UML modeling language, data storage is provided by the MySQL database. For the implementation, we used the Java language which allowed us to have a prototype to manage an election from its opening to its closing by allowing the sending of the information note to the voters, the applications, the choice of the candidate (by voting), the blocking of access after voting and the supervision of the polling statistics and the printing and sending by e-mail of the minutes of the results.

DEDICACES

C'est avec une profonde gratitude et sincères mots, que je dédie ce modeste travail de fin d'études à mes chers parents qui ont sacrifié leur vie pour ma réussite. Ils m'ont guidé et ils m'ont éclairé le chemin par leur conseil judicieux. J'espère qu'un jour, je pourrais leur rendre un peu de ce qu'ils ont fait pour moi, que Dieu leur accorde tout le bonheur du monde et longue vie.

Ce travail est dédié aussi à mes frères, sœurs et amis pour leur amour, leur soutien et leurs encouragements.

Merci à vous tous.

REMERCIEMENTS

Je ne saurais débuter ce mémoire sans remercier toutes les personnes de bonne volonté qui n'ont ménagé aucun effort pour la réalisation de ce modeste travail de mémoire de fin d'études.

Je tiens à remercier mon encadrant Mr Khadim pour son implication dans l'ensemble du travail. Vous avez consacré beaucoup de temps à m'apporter des outils et conseils indispensables à la bonne conduite du projet.

Je remercie particulièrement le Directeur des Ressources Humaines Mr Alain GOMIS et le Chef des services Pédagogiques de l'UFR des Science et Technologies Mr Alioune Badara Dieng, ils m'ont permis d'avoir le maximum d'informations nécessaires sur l'organisation des élections au sein de l'Université Assane SECK de Ziguinchor (UASZ). Ils se sont vraiment impliqués dans la phase d'enquêtes et ont ressenti ce besoin d'acquérir un système adéquat pour l'organisation d'élections à l'UASZ.

Un grand merci à la communauté estudiantine qui a manifesté un grand intérêt sur ce sujet. Ils ont exprimé leur envie d'un changement dans l'organisation des élections à l'UASZ.

Au terme de ce modeste travail, nous tenons aussi à remercier chaleureusement et respectivement tous ceux qui y ont contribué de près ou de loin, particulièrement Dr Youssou Faye, qui a permis pour une première fois la sélection en Master, d'étudiants venant de la Licence 3 d'Informatique Appliquée. Sa confiance en nous n'a jamais cessé d'accroître.

Des remerciements vont également à l'égard de tous les enseignants du département d'Informatique, les intervenants externes et l'équipe pédagogique pour toutes les connaissances théoriques et pratiques et l'accompagnement qu'ils m'ont apportés tout au long de mon cursus universitaire.

Enfin, je remercie mes amis et camarades de promotion pour ces années passées ensemble.

Merci à tous.

SOMMAIRE

RESUME i
ABSTRACT ii
DEDICACES iii
REMERCIEMENTS iv
LISTE DES FIGURES ix
LISTE DES TABLEAUX xi
LISTE DES ABBREVIATIONS xii
INTRODUCTION GENERALE 1
CHAPITRE I : CONTEXTE JUSTIFICATIF DU SUJET DE MEMOIRE 3
I. Etat de l'art sur les systèmes de vote électronique 3
I.1 Présentation des différents moyens de vote 3
I.2 Historique du vote électronique 4
I.3 Les environnements contrôlés et non contrôlés 5
a) Les environnements contrôlés 5
b) Les Environnements non contrôlés 5
I.4 Présentation de certains systèmes de vote par internet 6
a) Le système HELIOS 6
b) Le système BELENIOS 7
I.5 Les enjeux sur l'utilisation des systèmes de vote par internet 8
II. Présentation et étude des différentes élections organisées au sein de l'UASZ 8
II.1 Les différentes élections organisées à l'UASZ 9
a) L'élection de Vice-recteurs 9
b) L'élection des membres du CA 10
c) L'élection du Directeur d'UFR 10
d) L'élection du Chef de département 10
e) L'élection des délégués des étudiants 10
f) L'élection du président de l'amicale des étudiants 11
III. Problèmes rencontrés sur l'organisation des élections 13
III.1 Le temps et le travail fastidieux sur l'organisation des élections 13
III.2 Problème de transparence et de fiabilité 13
III.3 Le faible taux de participation 14
III.4 La participation simultanée à plusieurs élections 14

IV. Problématique du sujet 14
IV.1 Solution proposée 15
IV.2 Objectifs spécifiques 15
CHAPITRE II : CYCLE DE DEVELOPPEMENT DE L'APPLICATION 17
I. Présentation de quelques méthodes traditionnelles 17
I.1 Modèle en Cascade 17
I.2 Modèle en V 18
I.3 Modèle en Y 18
II. Présentation de quelques méthodes agiles 18
II.1 Rapid Application Development (RAD) 19
II.2 Unified Process (UP) 19
II.3 Extreme Programming (XP) 19
III Etude Comparative entre les méthodes traditionnelles et agiles 20
IV La Présentation du Framework SCRUM 21
V. Les cycles du Framework SCRUM 22
VI. Organisation de l'équipe avec le Framework SCRUM 23
1. L'organisation de l'équipe projet 23
2. Déroulement du projet sur SCRUM 24
CHAPITRE III : SPECIFICATION ET ANALYSE DES BESOINS FONCTIONNELS 25
I. Spécification des besoins 25
1. Identification des acteurs 25
2. Identification des fonctionnalités 26
3. Diagramme de cas d'utilisation 27
a) Diagramme de cas d'utilisation de l'administrateur 28
b) Diagramme de cas d'utilisation du superviseur 28
c) Diagramme de cas d'utilisation des électeurs et candidats 29
4. Description des cas d'utilisation 30
a. Description du cas d'utilisation « s'authentifier » 30
b. Description du cas d'utilisation « voir liste des candidats » 31
c. Description du cas d'utilisation « voir liste des électeurs » 31
d. Description du cas d'utilisation « visualiser le vote » 32
e. Description du cas d'utilisation « Envoyer note d'information » 32
f. Description du cas d'utilisation « valider candidature » 33
g. Description du cas « imprimer procès-verbal résultat » 33
h. Description du cas d'utilisation « Déposer candidature » 34

i. Description du cas d'utilisation « effectuer vote » 35
j. Description du cas d'utilisation « Paramétrer élection » 35
II. Analyse des besoins fonctionnels du système 36
1. Etude des activités de « déposer candidature » 36
a. Diagramme de séquence du cas « déposer candidature » 36
b. Diagramme d'activité du cas « déposer candidature » 37
2. Etude des activités du cas « effectuer vote » 38
a. Diagramme de séquence 38
b. Diagramme d'activité 38
3. Etude des activités de « visualiser vote » 39
a. Diagramme de séquence 39
b. Diagramme d'activité 40
CHAPITRE IV : CONCEPTION DU SYSTEME 42
I. Conception générale du système 42
1. Architecture du système 42
a. Modèle MVC 42
b. Architecture 3-Tiers 43
2. Diagramme de packages 45
3. Diagramme de déploiement 46
II. Conception détaillée du système 46
1. Description des différentes classes du système 47
2. Diagramme de classes du système 47
CHAPITRE V : IMPLEMENTATION ET PRESENTATION DE L'APPLICATION 49
I. Implémentation de l'application 49
1. Les serveurs utilisés pour notre application 49
a. SGBD (MySQL) 49
b. Serveur d'application (Apache Tomcat embarqué) 50
2. Les technologies utilisées à l'implémentation 50
a. Technologies utilisées pour la partie Back-end 50
b. Technologies utilisées pour la partie Front-End 52
3. Les outils utilisés pour l'implémentation 53
a. Intellij Idea 53
b. Star UML 53
c. L'outil Git 53
4. L'arborescence des fichiers de l'application 53

5. L'arborescence des packages de l'application 54
a. Les packages de la partie Back-end 55
b. Les packages de la partie Front-end 61
II. Présentation de l'application 61
1. Authentification 62
2. Menu Administration 62
3. L'ouverture des élections 63
4. La phase de dépôt de candidature 63
5. La phase de scrutin 64
6. La phase de supervision et de publication des résultats 66
CONCLUSION GENERALE ET PERSPECTIVES 68

LISTE DES FIGURES

Figure 1**: Système de vote électronique fixe** 5

Figure 2**: Principe de fonctionnement du système BELENIOS** 8

Figure 3**: Procédé de Fonctionnement du Framework SCRUM** 23

Figure 4**: Diagramme de cas d'utilisation de l'administrateur** 28

Figure 5**:Diagramme de cas d'utilisation du superviseur** 29

Figure 6: **Diagramme de cas d'utilisation électeur et candidat** 30

Figure 7**: Diagramme de séquence "déposer candidature"** 37

Figure 8**: Diagramme d'activité de "déposer candidature"** 38

Figure 9**: Diagramme de séquence "effectuer vote"** 38

Figure 10**: Diagramme d'activité "effectuer vote"** 39

Figure 11**: Diagramme de séquence "visualiser le vote"** 40

Figure 12: **Diagramme d'activité du cas "visualiser avancé du vote"** 41

Figure 13**: Présentation du modèle MVC** 43

Figure 14: **Architecture 3 Tiers** 44

Figure 15**: L'architecture de notre application** 44

Figure 16**: Diagramme de Packages** 45

Figure 17**: Diagramme de déploiement** 46

Figure 18**:Les différentes classes du système** 47

Figure 19**: Diagramme de classe détaillé de l'application** 48

Figure 20:**Spring boot par rapport à Spring** 51

Figure 21**: Arborescence des fichiers de l'application** 54

Figure 22**: L'arborescence des packages de l'application** 55

Figure 23**: Exemple de la classe "User"** 56

Figure 24**:Exemple d'interface "CandidatRepository"** 57

Figure 25**:exemple interface "IBulletin"** 58

Figure 26**:Exemple de la classe "BulletinImpl"** 58

Figure 27**:exemple de controller "BulletinController"** 59

Figure 28**: Exemple de la classe "SecurityConfig"** 60

Figure 29: **Vue globale de l'application** 62

Figure 30**: Page d'authentification du système** 62

Figure 31 **: Menu administrateur** 63

Figure 32**: Ouverture d'une élection** 63

Figure 33**: Formulaire de dépôt de candidature** 64

Figure 34**: Boîte d'option élection** 65

Figure 35**: Liste des bulletins de candidats** 65

Figure 36**: Message de confirmation** 65

Figure 37**: Menu électeur avec l'élection restante** 66

Figure 38: **Les résultats du scrutin** 66

Figure 39: **Procès-verbal du scrutin** 67

LISTE DES TABLEAUX

Tableau 1: **Tableau récapitulatif des élections organisées** 11

Tableau 2**: Tableau de comparaison entre la méthode traditionnelle et agile** 20

Tableau 3**: Organisation de l'équipe projet SCRUM** 23

Tableau 4**: Identification des acteurs** 26

Tableau 5**: Identification des fonctionnalités** 26

Tableau 6**: Description du cas d'utilisation "s'authentifier"** 30

Tableau 7**: Description du cas d'utilisation "voir liste des candidats"** 31

Tableau 8**: Description du cas d'utilisation "voir liste des électeurs"** 31

Tableau 9**: Description du cas d'utilisation "visualiser le vote"** 32

Tableau 10**: Description du cas d'utilisation "envoyer note d'information"** 32

Tableau 11**: Description du cas d'utilisation "valider candidature"** 33

Tableau 12**: Description du cas d'utilisation "imprimer procès-verbal des résultats"** 33

Tableau 13**: Description du cas d'utilisation "Déposer candidature"** 34

Tableau 14**: Description du cas d'utilisation "effectuer vote"** 35

Tableau 15**: Description du cas d'utilisation "paramétrer élection"** 36

LISTE DES ABBREVIATIONS

AOP:	Aspect Oriented Programming
API:	Application Programming Interface
CA :	Conseil d'Administration
CSS:	Cascading Style Sheets
CNRS:	Centre Nationale de la Recherche Scientifique
CAMES:	Conseil Africain et Malgache pour l'Enseignement Supérieur
CSA:	Chef de Service Administratif
CSP:	Chef de Service Pédagogie
DAO:	Data Access Object
DOM:	Document Object Model
DRH:	Direction des Ressources Humaines
GNU:	General Public Licence
HTML:	Extensible Hypertext Markup Language
HTTP:	HyperText Transfer Protocol
HTTPS:	Hypertext Transfer Protocol Secure
IDE:	Integrated Development Environment
INRIA:	Institut National de Recherche en Informatique et en Automatique
JPA:	Java Persistence API
JS:	JavaScript
JSP:	Java Server Page
MVC:	Model View Controller
MySQL:	My Structured Query Language
ORM:	Object Relational Mapping

PER:	Personnel Enseignant et de Recherche
PATS:	Personnel Administratif, Technique et de Service
POM:	Project Object Model
PV:	Procès-Verbal
RMI:	Remote Method Invocation
SGBDR:	Système de Gestion de Base de Données Relationnelles
SMS:	Short Message Service
SQL:	Structured Query Language
UASZ:	Université Assane SECK de Ziguinchor
UFR :	Unité de Formation et de Recherche
UML:	Unified Modeling Language
XML:	Extensible Markup Language

INTRODUCTION GENERALE

Pour faire régner la démocratie et la transparence, la société humaine a toujours trouvé les moyens nécessaires lui permettant de faire les choses de façon équitable et compréhensible par tout un chacun. Le vote s'est imposé comme un idéal en matière d'équité et de fiabilité.

Les votes peuvent être internes (concernant une organisation ou une institution) ou externes, c'est-à-dire à l'échelle régionale ou nationale (par exemple les élections communales ou présidentielles).

Les élections organisées à l'UASZ sont nombreuses et concernent les différents acteurs de l'institution. Chaque élection a ses règles et critères d'organisation bien définis.

Les votes sont faits aujourd'hui suivant le système traditionnel, plus précisément l'urne avec comptage manuel des voix.

Au sein de l'université, les votes se font en utilisant l'urne ; les électeurs se présentent au niveau d'une salle et procèdent au vote en déposant une enveloppe contenant le bulletin choisi au niveau de l'urne ; le dépouillement et la proclamation des résultats se font en présence des différents candidats ou de leurs représentants. Le vote par mail ou par procuration est également permis à ceux qui ne sont pas sur place.

Malgré leur caractère transparent, ces moyens de vote engendrent aussi certains problèmes. En effet, la gestion, ainsi que l'organisation d'une élection peuvent s'avérer fastidieuses. Beaucoup de temps sont perdus sur l'organisation et la communication sur l'ouverture du scrutin n'est pas optimale ; ce qui peut, en conséquence, jouer sur le taux de participation. Le comptage manuel des voix est certes sûr, mais rien ne garantit l'absence d'erreurs de comptage.

Un autre procédé de vote est aussi utilisé à l'UASZ, c'est le vote par correspondance qui peut aussi présenter certains problèmes en termes de transparence et de fiabilité.

Face à ces problèmes, nous nous sommes intéressés aux différents modes de scrutin de l'UASZ et avons réfléchi sur des solutions permettant de les alléger.

Nous avons ainsi proposé la mise en œuvre d'un système de vote électronique en ligne permettant d'assurer de façon efficace l'organisation et le déroulement d'élections tout en garantissant la sécurité et la transparence.

Pour réaliser les fonctionnalités attendues, nous avons fait appel aux principes et démarches du génie logiciel en faisant notre choix sur un processus de développement bien précis tel que SCRUM. Pour l'analyse et la conception, nous avons utilisé le langage de modélisation UML. Nous nous sommes focalisés sur l'organisation et le déroulement des votes en prenant en compte les acteurs concernés et les critères définis pour chaque élection.

Notre mémoire est subdivisé en cinq chapitres :

- Présentation et contexte justificatif du sujet de mémoire ;
- Cycle de développement de l'application ;
- Spécifications et Analyse des besoins ;
- Conception du système ;
- Implémentation et présentation de l'application

CHAPITRE I : CONTEXTE JUSTIFICATIF DU SUJET DE MEMOIRE

Ce chapitre, permet de délimiter l'étude du sujet et de montrer clairement nos motivations. Dans cette partie, nous allons aborder l'étude sur trois points. Premièrement, nous ferons l'état de l'art sur les différents systèmes de vote existant ; ensuite, nous ferons une présentation et une étude sur l'organisation d'élections au sein de l'UASZ ; enfin, nous aborderons les problèmes rencontrés.

I. Etat de l'art sur les systèmes de vote électronique

À partir du milieu des années 90, les modalités de votes subissent la grande créativité. Dès lors, sont promues deux tendances qui proposent une informatisation différente des modes de scrutin. Pour la première, le vote électronique signifie l'intégration *via* l'« urne électronique » (aussi appelée « machine à voter » dans le droit français) de procédés permettant de faire intervenir des entreprises privées dans le système de vote. L'argument commercial, principal utilisé pour promouvoir ces produits repose sur l'idée d'accélérer le processus de traitement des suffrages exprimés. Pour la deuxième, l'informatisation du processus de vote permettra de voter à distance. C'est-à-dire de voter de chez soi, ou de n'importe où dans le monde, et ainsi éviter de se déplacer dans des bureaux de vote, mais sans aucune garantie de l'individu qui vote. Toutefois, pour des enjeux importants, notamment politiques, le vote électronique pose des problèmes de vérification des votes individuels.

I.1 Présentation des différents moyens de vote

Trois types de vote sont utilisés que sont :

- **Le vote traditionnel manuel** :

  Le vote traditionnel a toujours existé et est toujours utilisé. Il requiert la présence de l'électeur au niveau du bureau de vote muni de sa carte d'identité et faisant le processus de vote avec une liste de choix qui lui est dressée. Le système classique de ce type est l'isoloir ou les électeurs vont procéder à faire le choix de leur candidat. Des pays comme les Etats-Unis ont mis en place des cabinets ou points de vote permettant une amélioration du processus.

- **Le vote mécanique :**

  Les dispositifs de vote mécaniques sont tous utilisés en environnement contrôlé. Le premier système de vote mécanique a été inventé en 1910 par Eugenio Boggiano. Ils

peuvent être classés en deux catégories selon que le bulletin de vote est dématérialisé ou pas. Pour le bulletin de vote dématérialisé, chaque électeur exprime son suffrage en appliquant une force motrice sur une pièce (bouton, levier) qui, par le jeu d'autres éléments mécaniques, a pour résultat d'incrémenter un compteur correspondant au choix initial. Pour le bulletin de vote matérialisé, chaque électeur s'exprime *via* un dispositif mécanique qui perfore une carte en un emplacement correspondant à son choix. Un second dispositif, électromécanique ou électronique, réalise le comptage des voix en parcourant l'ensemble des cartes perforées qui lui sont remises.

- **Le vote électronique :**

  Il existe une grande variété de dispositifs de vote électronique, mais tous font intervenir des traitements fondés sur l'électronique lors d'une ou de plusieurs étapes du processus de vote : expression du vote, émargement, transmission des suffrages, comptage des voix. Certains sont utilisés en environnement contrôlé, d'autres en environnement non contrôlé (comme internet, dans ce dernier cas, ils réalisent obligatoirement la gestion des émargements.). Enfin, les bulletins de vote connaissent des situations très contrastées selon qu'ils sont dématérialisés, ou matérialisés.

De ces différents moyens utilisés, notre étude va se pencher essentiellement sur le vote électronique, sur son historique, sur sa mise en application et aussi sur certains problèmes existants quant à son utilisation.

I.2 Historique du vote électronique

Les scanners sont utilisés pour la première fois en 1962 à Kern City, en Californie. Ils constituent la plus ancienne forme de vote électronique. Les électeurs s'expriment en cochant les cases de leur choix sur le bulletin habituel. À la fin du vote, les bulletins nourrissent un scanner qui fournit le décompte des suffrages.

Les ordinateurs de vote apparaissent en 1975 aux Pays-Bas et se répandent en Inde, au Brésil, en Belgique, aux États-Unis, et marginalement en France à partir de 2004. Pour exprimer leur choix, les électeurs interagissent directement avec le dispositif *via* des boutons physiques, un écran tactile, ou encore un pavé numérique. Les suffrages sont enregistrés directement dans une mémoire. À la clôture de la période de vote, le dispositif fournit les résultats électoraux par affichage ou impression d'un ticket. Certains ordinateurs de vote transmettent également leurs résultats vers le bureau centralisateur.

Le concept d'ordinateur de vote avec trace papier émerge en 2000 afin de pallier l'impossibilité de vérifier les résultats émis par un ordinateur de vote : lorsque l'électeur a confirmé les choix

qu'il a émis, l'ordinateur imprime un ticket portant les choix exprimés. Ce bulletin, dont l'électeur peut vérifier la justesse, est ensuite dirigé vers une urne à des fins d'éventuelles vérifications des résultats.

Depuis donc les années 2000, les systèmes de vote ont connu pas mal d'évolution. Néanmoins, chaque système fonctionne selon un environnement contrôlé ou non contrôlé **[1].**

I.3 Les environnements contrôlés et non contrôlés

a) Les environnements contrôlés

Les systèmes conçus dans ce type d'environnement sont contrôlés. Le système de vote est fixe et cela requiert la présence physique de l'électeur. Le processus de vote et le choix de l'électeur sont enregistrés sur un serveur central sécurisé fonctionnant sur un réseau privé qui est difficile à attaquer. Le choix de l'électeur ne peut donc être dévié par des tiers malintentionnés.

Ces systèmes sont des ordinateurs fixes présents dans les bureaux de vote et qui gèrent tous les processus de vote. Ce type de systèmes est illustré à la figure 1.

Figure 1: ***Système de vote électronique fixe***

b) Les Environnements non contrôlés

Certains systèmes fonctionnent dans un environnement non contrôlé, c'est-à-dire dans des environnements qui ne sont pas sujets à des contrôles de sécurité sur le processus électoral.

Objet usuel de communication orale, le téléphone peut être utilisé en situation de vote. L'électeur appelle une ligne spécialement dédiée à l'élection, il s'identifie (donne son identité.), la preuve par un mot de passe (envoyé au préalable par courrier), puis exprime son choix en utilisant les touches numériques du téléphone. Un système de centralisation recueille les votes puis présente des résultats à la clôture de la période de vote. Cette procédure peut être aménagée dans le cas où l'on utilise le Short Message Service (SMS). L'électeur reçoit au préalable un

code pour s'identifier et la liste des chiffres lui permettant d'exprimer son choix. Durant la période de vote, il s'exprime en envoyant un SMS contenant ce code et le chiffre correspondant à son choix.

Nous avons aussi le vote par Internet où les électeurs procèdent au vote depuis n'importe quel ordinateur connecté à la toile ; la contrainte géographique est absente. Il faut se connecter sur le site officiel de vote, site hébergé sur l'ordinateur du bureau centralisateur (le serveur). Le vote se déroule alors plus ou moins comme sur un kiosque.

Actuellement, beaucoup de systèmes de vote fonctionnent sur un environnement non contrôlé ; le principal défi reste la question de la sécurité et de la transparence des votes. Le vote par internet est le plus utilisé. Diverses applications ont été conçues pour favoriser ce type de vote **[1].**

I.4 Présentation de certains systèmes de vote par internet

Il existe un grand nombre de systèmes de vote électronique utilisé. Les critères d'évaluation de chaque système résident sur l'étude de certains points essentiels tels que la sécurité de vote, l'authenticité de l'électeur, l'efficacité sur le processus de vote et de dépouillement des bulletins. Des systèmes comme BELENIOS et HELIOS sont des exemples typiques des nouveaux systèmes de vote électronique mis en place et qui suscitent notre intérêt pour une étude **[1].**

a) Le système HELIOS

HELIOS est un système de vote électronique dont l'objectif est de garantir deux points essentiels dans la notion de vote : la confidentialité et la vérifiabilité. Il a été développé en 2008. C'est un logiciel libre sous la licence Apache 2.0.

Sur l'ordinateur client, l'électeur vote en choisissant son candidat. Ce vote est crypté avec une clé de chiffrement publique. Après identification de l'électeur par un couple identifiant/mot de passe, le vote est envoyé sur le serveur de vote. Le vote est alors « visible » sur une page web où une urne publique est présente permettant de vérifier son vote.

Le dépouillement nécessite une clé de déchiffrement privée partagée avant le vote entre trois ou quatre autorités de déchiffrement (maire, responsables de différents partis, …). Chacune de ces parties possède une « partie » de la clé de sorte qu'au moins deux parties de la clé soient nécessaires pour déchiffrer le résultat de l'élection.

Les limites avec ce type de systèmes sont identifiées ; si l'ordinateur de l'électeur est infecté par un virus, il peut altérer le vote avant son chiffrement. Des solutions sont à l'étude reposant sur

un autre appareil, tel qu'un téléphone portable permettant de vérifier le vote reçu par l'urne. Des tentatives d'attaques ont été notées en cours de son utilisation.

Un autre problème, c'est que le système n'a aucune résistance à la coercition. Certains experts indiquent ainsi qu'un tel système **ne doit pas être utilisé** dans un vote à fort enjeu où l'achat de vote est possible (par exemple des élections présidentielles). La question de la transparence est donc remise en question **[1].**

b) Le système BELENIOS

BELENIOS constitue une évolution du système HELIOS. BELENIOS est développé sous licence libre par Stéphane GLONDU (ingénieur de recherche INRIA), en collaboration avec Pierrick Gaudry (directeur de recherche CNRS).

BELENIOS a été conçu pour éviter les risques de bourrage d'urne même pour un attaquant contrôlant le serveur de vote.

Dans son principe de fonctionnement, au début de l'élection, chaque électeur reçoit les éléments nécessaires pour voter, à savoir un login et un mot de passe pour s'authentifier, ainsi qu'un autre identifiant privé, appelé jeton de vote, qui permettra à l'électeur de prouver qu'il a bien le droit de voter. Ce jeton de vote est en fait une clé de signature.

La phase de vote est représentée à la figure ci-dessous. À chaque élection, est associée une clef publique (notée pub. E/ dans la figure). Le chiffrement à clef publique est un système de chiffrement dit asymétrique : si la clef de chiffrement est publique et tout le monde peut chiffrer, alors que si la clef de déchiffrement est privée, seules les personnes ayant la clef de déchiffrement peuvent déchiffrer. Pour voter, l'électeur, via son navigateur, chiffre son choix vD (0 ou 1) avec la clef publique de l'élection. Ce chiffrement est noté fvDgpub.E/. Il fournit également la preuve qu'il a bien chiffrée l'une des deux valeurs 0 ou 1 et non une autre valeur. Le vote chiffré, accompagné de la preuve de validité, est signé par le jeton de vote envoyé préalablement à l'électeur. L'ensemble forme le bulletin, noté [fvDgpub.E/] jeton D dans la figure ci-dessous.

Pour la phase de dépouillement, nous avons une clé de déchiffrement subdivisée et confiée à chaque autorité. Chacune des autorités de déchiffrement contribue au déchiffrement du résultat chiffré.

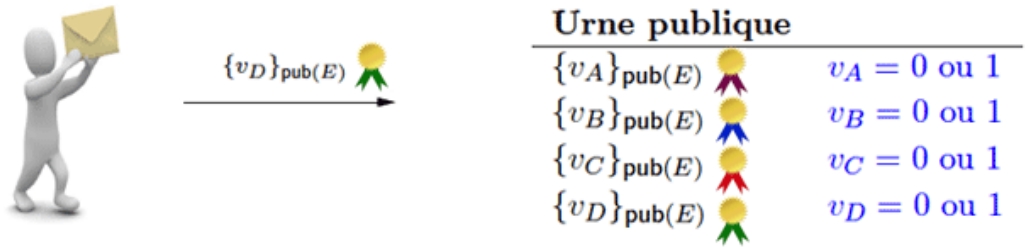


Figure 2: ***Principe de fonctionnement du système BELENIOS***

BELENIOS, comme tout système de vote en ligne, ne semble pas adapté à des élections à forts enjeux, comme des élections politiques (présidentielles, législatives, primaires...). Les risques d'attaques deviennent accrus. Son ergonomie et son principe d'utilisation peuvent être difficiles à comprendre pour un utilisateur lambda. D'ailleurs, même son mode de fonctionnement a toujours été critiqué **[1].**

I.5 Les enjeux sur l'utilisation des systèmes de vote par internet

La grande question qui subsiste toujours sur l'utilisation des systèmes de vote en ligne est l'assurance totale de la sécurité et de la transparence de vote. L'utilisation de ces types de système reste toujours critiquée par des experts qui jugent que les risques d'attaque sont énormes. Il y a aussi l'utilisation qui peut s'avérer très délicate. Certains de ces systèmes nécessitent une petite formation à un mois des échéances électorales. Ce qui peut s'avérer très pénible en termes de coût et de ressources.

Malgré toutes ces limites, nous considérons qu'ils sont efficaces dans la mesure où le processus de vote devient rapide et efficace. On perd moins de temps dans l'organisation et le comptage des voix et le délai de proclamation des résultats est court. Les risques d'erreur de comptage des voix sont inexistants. Pour renforcer les moyens de sécurité, on peut faire appel à des techniques de cryptographie modernes très puissantes ou même utiliser des serveurs proxy hautement efficace pour garantir la sécurité. Pour des élections à faibles enjeux, les risques d'attaques sont minimes et ils pourront être utilisés en intranet au sein d'une organisation (tels que les entreprises, Universités, etc.) **[1].**

II. Présentation et étude des différentes élections organisées au sein de l'UASZ

L'Université Assane SECK de Ziguinchor (UASZ) est créée par le décret 2008-537 du 22 mai 2008. Elle a démarré ses activités en février 2007 avec (03) Unités de Formation et de Recherche (UFR) et des Centres et Services communs. Elle a connu pas mal d'avancée et compte actuellement quatre UFR et dans chaque UFR, nous avons des départements et filières. Ces UFR sont :

- UFR des Sciences et Technologies(ST) ;
- UFR Sciences Economiques et Sociales(SES) ;
- UFR des Sciences de la Santé(SS) ;
- UFR des Lettres, Arts et Sciences Humaines (LASHU).

A côté de ces UFR, nous avons un ensemble de services administratifs et techniques qui veillent au fonctionnement interne comme externe de l'Université.

Chaque UFR et service sont dirigés par une personne occupant ainsi un poste bien défini dans la hiérarchie organisationnelle de l'UASZ.

Dans l'organisation de l'Université, nous avons le Recteur qui coiffe le haut rang, suivi des Vice-recteurs, des Directeurs d'UFR, des Chefs de département, des Directeurs et Responsables de services (CRI, Bibliothèque, Scolarité, etc.).

Certains de ces postes sont à caractère nominatifs (par exemple, le Recteur est nommé par décret présidentiel.). D'autres, par contre, font l'objet d'élections internes organisées au sein de l'université (Chef de département, Directeur d'UFR, Vice-recteur). À côté de ces élections, nous en avons d'autres à caractère représentatif d'une entité (par exemple celle des délégués). Chaque élection a son collège électoral. Le collège électoral constitue les règles et conditions pour la participation des membres à l'élection. Par exemple, le collège électoral de l'élection du vice rectorat (membre de conseil d'administration et professeur de rang A) est différent de celui de l'élection des délégués d'étudiants (étudiant de master 2 et de Licence 3 non-éligible) Ces différentes élections sont décrites dans les sections qui suivent.

II.1 Les différentes élections organisées à l'UASZ

a) L'élection de Vice-recteurs

Ce scrutin concerne les deux postes de Vice-recteurs : Vice-recteur chargé des Études et de la Vie de l'Université (EVU) et Vice-recteur chargé de la Recherche, de la Coopération et des relations avec le monde professionnel. C'est une élection très fermée ; le collège électoral est constitué des membres du conseil d'administration (CA) ; le candidat doit être un enseignant de rang A (Professeur titulaire ou Maître de Conférence CAMES) et membre du CA.

Dans ce scrutin, le vote par procuration est permis, c'est-à-dire qu'on peut voter pour une personne absente s'il nous délègue cette tâche. Cette élection se fait sous la supervision de la direction des ressources humaines (DRH).

b) L'élection des membres du CA

Cette élection permet aux PER, PATS ainsi qu'aux étudiants de désigner respectivement leurs représentants au CA. Les enseignants de rang A choisissent 7 membres parmi eux, les enseignants de rang B 14 parmi les pairs, le PATS choisit 2 et les étudiants auront 4 membres (c'est-à-dire les présidents élus des différentes UFR) au conseil d'administration. Cette élection est un peu particulière ; les électeurs peuvent voter pour autant de candidats possibles selon le nombre de sièges à pourvoir ; de même, chaque électeur ne peut voter que pour la catégorie à laquelle il appartient.

Les syndicats des enseignants et du personnel administratif désignent également leurs représentants au CA. Ces choix se font d'interne.

Le vote par correspondance de même que le vote par envoi de courrier électronique sont acceptés.

c) L'élection du Directeur d'UFR

Le Directeur d'UFR dirige son UFR du point de vue managérial. Il est élu par les enseignants et le PATS de l'UFR. Pour concourir à ce poste, il faut être un enseignant de rang A (c'est-à-dire Professeur titulaire ou Maître de Conférences CAMES). L'élection est sous la supervision du Vice-recteur EVU. Le vote par correspondance est permis dans ce type de scrutin.

d) L'élection du Chef de département

Dans chaque département, on a un responsable qui gère le fonctionnement administratif et pédagogique. Le chef de département est élu par le PER rattaché au département. Par exemple, pour le département d'Informatique, le personnel enseignant choisit son chef par vote. Le vote par correspondance est permis. Cette élection est organisée sous la supervision du Directeur et du Chef des Services Administratifs (CSA) de l'UFR.

e) L'élection des délégués des étudiants

Les délégués représentent les étudiants à l'amicale, mais aussi au conseil d'administration. Chaque délégué est élu par les étudiants de sa propre classe. Pour l'élection des délégués, c'est l'administration chef des Services Pédagogiques (CSP) qui gère l'ensemble du processus.

Avant que les élections ne soient lancées, l'UFR affiche un communiqué d'ouverture des candidatures pour les étudiants voulant être délégués. Puis, chaque candidat se présente à l'UFR auprès du CSP pour déposer sa candidature. Un autre facteur doit être pris en compte : si pour une classe, on a un effectif inférieur ou égal à 50 étudiants, elle aura un seul délégué ; si l'effectif est supérieur à 50 étudiants, elle aura deux délégués. Si pour une classe, on a une seule candidature, le vote ne sera pas nécessaire, car la classe a donné son consentement.

Pour le déroulement, l'UFR et la classe concernée vont trouver des heures creuses pour procéder au vote. Le CSP et son équipe vont venir à la classe et superviser le vote. Les étudiants vont faire leur choix et les déposer dans l'urne. Une fois le vote fini, le CSP procède au dépouillement et proclame ensuite les résultats.

f) L'élection du président de l'amicale des étudiants

Une fois que l'élection des délégués est terminée, nous allons donc procéder à l'élection du président du bureau de l'amicale des étudiants. Certains critères sont exigés à ce niveau ; le président ne doit pas être un étudiant de première année, ni un étudiant en position de sortie. Le président élu va représenter les étudiants au niveau du conseil d'administration de l'Université. L'élection du président ne concerne que les délégués élus. Ce sont eux qui vont se charger d'élire le président sous la supervision du directeur de l'UFR et de son CSA. Il faut préciser que pour chaque UFR, on a un président d'amicale.

Nous en avons d'autres élections, mais celles énumérées sont les plus importantes. La direction des ressources humaines supervise un grand nombre de ces élections.

Toutefois, dans ces élections, différents problèmes sont rencontrés et suscitent un intérêt pour notre travail. Ci-dessous, nous avons un tableau récapitulatif de l'ensemble des élections organisées à l'Université.

Tableau 1: ***Tableau récapitulatif des élections organisées***

Elections	Descriptions	Méthodes de vote
Election Vice-Recteur chargé des études et de la vie de l'Université (EVU)	Cette élection permet donc la nomination du Vice-Recteur pour un mandat de 3 ans. Collège électoral constitué des candidats de rang A et membres du CA	Vote par Urne et sous la supervision du directeur des ressources humaines. Le vote peut se faire aussi par mail ou par procuration.

Election Vice-Recteur chargé de la recherche, la coopération et les relations avec le monde professionnel	Cette élection permet la nomination du Vice-Recteur pour un mandat de 3 ans. Collège électoral constitué des candidats de rang A et membres du CA	Vote par Urne et sous la supervision du directeur des ressources humaines. Le vote peut se faire aussi par mail ou par procuration.
Election des membres au conseil d'administration	Cette élection vise à la nomination des membres au conseil d'administration. Le collège électoral est constitué de candidats représentant respectivement PER, PATS.	Vote par Urne et sa particularité est qu'on peut voter pour plusieurs candidats à la fois. Elle est sous la supervision du DRH
Election du directeur d'UFR	Elle permet la nomination du directeur UFR pour un mandat de 3 ans. Le collège électoral est constitué des candidats de rang A de l'UFR	Vote par urne et sous la supervision du Vice-Recteur (EVU)
Election du chef de département	Elle permet la nomination du chef de département rattaché à l'UFR pour un mandat de 3 ans. Le collège électoral est constitué des candidats enseignants du même département.	Vote par urne et sous la supervision du directeur UFR et du CSA. Vote par procuration est possible.
Election des délégués des étudiants	Elle permet la nomination des délégués de l'amicale des étudiants pour un mandat de 1 an.	Vote par urne et sous la supervision du chef du service pédagogie (CSP)
Election du président de l'amical des délégués étudiant	Elle permet la nomination du président de l'amicale pour un mandat de 1 an. Le Collège électoral est	Vote par urne et sous la supervision du directeur d'UFR et du CSA.

	constitué par les candidats membres de l'amicale.	

III. *Problèmes rencontrés sur l'organisation des élections*

L'organisation d'une élection peut s'avérer très pénible. Les élections organisées à l'UASZ n'en font pas exception. Certains problèmes sont donc visibles sur l'organisation des votes et peuvent constituer des points majeurs soulevés. Les problèmes les plus marquants sont : le temps et le travail fastidieux, le taux de participation faible, la participation simultanée à plusieurs élections.

III.1 Le temps et le travail fastidieux sur l'organisation des élections

D'après le CSP, il consacre beaucoup de temps à l'organisation des élections des délégués. Le recensement des candidats ainsi que la vérification de leur éligibilité leur coûtent beaucoup de temps ; lui et son équipe doivent vérifier pour chaque candidat les critères de sélection.

Pendant la phase du vote, ils doivent faire le tour de chaque classe pour procéder à l'élection ; ceci a un impact en termes de ressources. Par exemple pour les étudiants se trouvant à l'EATEA, il faudra que le CSP se déplace jusque là-bas et ainsi procéder au vote. Et plus encore, ils doivent attendre à des heures creuses ou bien même interrompre des cours afin de procéder à l'élection. L'organisation des élections des délégués est très pénible et nécessite beaucoup de ressources et de travail fastidieux de la part de l'administration.

De même que pour les autres élections, l'organisation peut prendre beaucoup de temps. Il arrive qu'il y ait décalage de la date du scrutin, car le nombre de candidats inscrits est faible. Et dans la phase de proclamation, le dépouillement et le comptage des voix prennent du temps et les risques d'erreur (erreur de comptage), néanmoins faibles ne sont pas à négliger.

III.2 Problème de transparence et de fiabilité

Le principe d'une élection est que les électeurs ainsi que les candidats acceptent de se conformer aux résultats. Il faudra donc faire confiance en la sincérité du scrutin. Le problème de transparence repose surtout sur le dépouillement et la proclamation des résultats. Le vote à l'urne semble être transparent, mais peut-on se fier aux personnes faisant le dépouillement des votes. À travers mon entretien avec le CSP, il parlait de la transparence du vote, c'est-à-dire que même les votes pouvaient être truqués par ceux qui l'organisent dans la mesure où ces derniers pouvaient falsifier les résultats des votes. Ce n'est pas une affirmation de sa part, mais

c'est une possibilité. Ce qui constitue un risque non-négligeable. La question de la fiabilité du vote repose donc essentiellement sur les acteurs impliqués dans l'organisation. La neutralité des superviseurs doit impérativement être prise en compte, car si elle est négligée cela peut être un risque énorme au niveau du dépouillement.

D'après le Directeur des ressources humaines (DRH), le vote par e-mail n'est pas très fiable. En effet, celui qui vote par exemple via son adresse e-mail de l'Université peut aussi voter via une autre adresse e-mail du fait qu'il n'y a pas un contrôle pointu fait à ce niveau. La vérification sur l'authenticité de l'électeur est presque absente ; ce qui rend vulnérable la sécurité du vote.

Nous avons aussi un autre problème lié au vote par procuration, où un électeur, peut décider de déléguer son vote à un mandataire pour qu'il le fasse à sa place. La confiance totale n'est pas assurée car ce même mandataire peut être tenté de modifier le choix de vote défini au départ par l'électeur mandant. Le vote par procuration nécessite une attention particulière car sa transparence est toujours remise en question.

III.3 Le faible taux de participation

D'après le DRH, le taux de participation aux élections est minime du fait que la communication ne passe pas très bien. Le nombre de participants aux élections est inférieur au nombre moyen qui doit participer. À l'ouverture d'une élection, ils affichent une note d'information. La note d'information n'est pas prise en compte et en conséquence le nombre de candidats et d'électeurs est minime. La couverture médiatique n'est pas optimale.

L'absence des participants a un impact aussi sur le taux de participation. En effet, beaucoup d'électeurs partent en voyage et par conséquent, ils ratent la phase de scrutin. Le faible taux de participation influe considérablement sur le suffrage valablement exprimé.

III.4 La participation simultanée à plusieurs élections

Plusieurs élections sont organisées simultanément à l'université. Les électeurs participent à la fois sur les élections ouvertes. Le problème est, qu'ils peuvent rater certains scrutins pour donner la priorité à d'autres et cela a un impact au niveau du taux de participation. Les électeurs devront être à mesure de participer simultanément à l'ensemble des élections ouvertes.

IV. Problématique du sujet

Depuis très longtemps, l'organisation des élections se faisait dans de telles conditions. Le vote manuel, malgré ces bénéfices, est très pénible dans son déroulement ; une institution comme l'Université a besoin de s'acquérir de moyens plus performants et rapides favorisant pleinement la gestion des élections. Notre application essayera d'apporter des améliorations dans l'organisation et le déroulement des votes tout en garantissant un accès sécurisé.

IV.1 Solution proposée

La solution proposée doit être conforme aux besoins et spécifications fonctionnelles définies. La plateforme devra permettre :

- **L'automatisation de l'organisation et du déroulement d'une élection** : tout le processus du vote doit être automatisé et la solution proposée doit être adaptée à tout type d'élection organisée à l'UASZ (délégués, Chef de département, Directeur d'UFR, …) ;
- **Les dépôts de candidature en ligne** : à partir de la plateforme, les utilisateurs peuvent déposer leur candidature sur une élection ouverte ;
- **La publication de la liste des candidats et du résultat de scrutin** : après chaque dépôt de candidatures, la liste des candidats retenus doit être imprimée et publiée. De même que les résultats à la fin du scrutin ;
- **L'envoi de notifications par mail sur l'ouverture et la publication des résultats** : à travers la plateforme, on peut envoyer des notifications par mail pour alerter les électeurs sur l'ouverture, le dépôt des candidatures et la publication des résultats.
- **Gérer simultanément plusieurs élections ouvertes** : Les élections ouvertes vont se dérouler en même temps sur la plateforme. Les électeurs concernés peuvent participer sur une ou plusieurs élections (uniquement sur les élections dont ils ont accès).

Dans sa conception, l'application va s'appuyer sur la liste des électeurs initialement connue, c'est-à-dire ; il n'y aura pas de procédure d'inscription des électeurs. Les électeurs sont juste ceux qui possèdent une adresse mail de l'université et qui appartiennent à l'entité concernée. Il faut aussi préciser que dans son fonctionnement, la plateforme peut gérer plusieurs élections ouvertes à la fois.

IV.2 Objectifs spécifiques

Les objectifs de notre application sont de :

- Réduire le temps de travail sur l'organisation, la supervision du processus de vote et le dépouillement des voix (qui se fera automatiquement) ;
- Gérer la fiabilité et la transparence du vote en assurant la vérification et la confidentialité du vote ;
- Augmenter le taux de participation à l'élection en envoyant des notifications par mail aux acteurs (candidats éligibles, électeurs, superviseurs) de chaque élection organisée ;
- Gérer simultanément plusieurs élections ouvertes au niveau de la plateforme ;

- Supprimer le vote par correspondance en permettant la mise en place d'un système décentralisé favorisant le vote pour toutes les personnes concernées ;

- Favoriser la sécurité du vote tout en gérant le contrôle d'accès à la plateforme selon le profil d'utilisateur ;

- Permettre la publication des listes imprimées des candidats retenus et des résultats des scrutins de vote.

CHAPITRE II : CYCLE DE DEVELOPPEMENT DE L'APPLICATION

Un processus ou cycle de développement est un ensemble d'activités ou étapes organisées à suivre pour réaliser et atteindre les objectifs fixés sur un projet. Il existe un grand nombre de méthodes et l'utilisation de chacune d'elle varie selon la complexité du projet en question. On distingue ainsi deux grandes méthodes que sont : les méthodes traditionnelles et les méthodes agiles.

Les méthodes agiles dans leur fonctionnement favorisent une meilleure compréhension des objectifs à atteindre sur un projet. Elles permettent ainsi un développement incrémental selon les nouvelles exigences des clients. Un suivi permanent est donc établi sur tout le processus de développement. La relation fluide avec le client est fortement conseillée. L'équipe de développement est organisée et hiérarchisée et les tâches définies pour chacun. Des réunions récurrentes sont effectuées pour évaluer l'avancée du projet.

Dans cette partie, notre intérêt porte essentiellement sur l'utilisation de la méthode SCRUM pour la réalisation de notre projet.

Elle est une des nombreuses méthodes agiles, existantes. Elle permet le suivi en permanence de tout le processus de développement et permet d'intégrer de nouveaux besoins pris en compte dans le cahier de charge.

Dans ce chapitre, nous allons d'abord faire une étude comparative entre les méthodes traditionnelles et agiles, ensuite présenter le Framework SCRUM et enfin essayer de l'adapter à notre cycle de développement.

I. Présentation de quelques méthodes traditionnelles

Dans cette partie, nous allons présenter quelques méthodes traditionnelles de gestion de projet.

I.1 Modèle en Cascade

Ce modèle est créé en Winston Royce en 1970 et il est basé sur deux éléments fondamentaux à savoir :

On ne peut passer à l'étape suivante tant que la précédente n'est pas terminée. Des validations sont faites dans chaque phase du cycle de développement (spécification, conception générale, conception détaillée, codage, intégration, mise en production, maintenance.).

La modification faite sur une étape va impacter sur l'ensemble du projet. Chaque modification doit être minutieusement établie afin d'éviter de grands changements sur le processus de développement.

Avec le modèle en cascade, le planning est établi à l'avance et le chef de projet sait précisément ce qui va lui être livré et quand il pourra en prendre livraison.

L'inconvénient du modèle en cascade est principalement sa faible tolérance à l'erreur (les anomalies sont détectées tardivement) qui induit automatiquement un coût important en cas d'anomalie.

I.2 Modèle en V

Nous avons vu précédemment le modèle en cascade, ce système imparfait a fait apparaître dans les années 80 un nouveau modèle qui est celui en V.

La caractéristique principale de ce processus est que chaque étape de conception fonctionne en binôme avec une phase de test (validation). Ensuite, plus on avance dans **l'étude** plus le niveau de détails est important.

Les étapes de tests sont aussi nombreuses que les étapes de conceptions et réflexions, ce qui permet un risque d'erreur moins élevé. Ainsi, le cahier des charges sera d'autant plus respecté. Malheureusement, ce modèle est rarement utilisé tel quel et le V est bien souvent déséquilibré, tantôt côté analyse, tantôt côté recette et la marge d'erreur est bien souvent proportionnelle à la marge de liberté prise par rapport au modèle théorique.

I.3 Modèle en Y

La réalisation du système consiste à fusionner les résultats des évolutions fonctionnelles et techniques pour former ainsi un processus de développement en Y. Les étapes fonctionnelles (analyse et conception) sont faites en parallèle avec les étapes techniques (codage). Le grand avantage du modèle est de permettre la réutilisation des modèles. Des modèles bien faits peuvent être réutilisés à chaque fois dans d'autres projets. Il suffira juste de les adapter sur le contexte du projet en cours.

Le principal problème du cycle en Y est la difficulté d'adaptation sur un projet. Son utilisation requiert une certaine maîtrise et connaissance de gestion de projet.

Il existe un grand nombre de méthodes traditionnelles dont chacune a ses propres particularités.

II. Présentation de quelques méthodes agiles

II.1 Rapid Application Development (RAD)

La méthode de **développement rapide d'applications**, dite **méthode RAD** (acronyme de l'anglais *Rapid Application Development*), est la première méthode de développement de logiciels où le cycle de développement est en rupture fondamentale par rapport à celui des méthodes antérieures dites « en cascade ». Ce nouveau cycle qualifié d'itératif, d'incrémental et d'adaptatif, se retrouvera dans toutes les méthodes dites « agiles » publiées par la suite.

La méthode RAD, après deux courtes phases de formalisation structurée de l'expression des besoins (CADRAGE) et de définition globale de l'architecture technique (DESIGN), inclut dans sa phase principale (CONSTRUCTION) la réalisation, la validation immédiate et les tests d'une application en mode itératif-incrémental-adaptatif. L'objectif de la méthode, qui implique activement l'utilisateur final dans un principe de « validation permanente », est d'obtenir un applicatif en adéquation avec les réels besoins.

II.2 Unified Process (UP)

Le **processus unifié** PU, ou **UP** (anglais : **Unified process**) est une méthode de développement pour les logiciels orientés objets. C'est une méthode générique, itérative et incrémentale, contrairement à la méthode séquentielle Merise (ou SADT).

Le processus unifié vient compléter le systémique des modèles UML. Elle est le résultat final d'une évolution de l'approche d'Ericsson qui est au fondement d'une des premières méthodes de développement pour applications orientées objets.

II.3 Extreme Programming (XP)

Extreme Programming (**XP**) est une méthode agile plus particulièrement orientée sur l'aspect réalisation d'une application, sans pour autant négliger l'aspect gestion de projet. XP est adapté aux équipes réduites avec des besoins changeants. XP pousse à l'extrême des principes simples. L'Extreme Programming repose sur des cycles rapides de développement (des itérations de quelques semaines) dont les étapes sont les suivantes :

- Une phase d'exploration détermine les scénarios « client » qui seront fournis pendant cette itération ;
- L'équipe transforme les scénarios en tâches à réaliser et en tests fonctionnels ;
- Chaque développeur s'attribue des tâches et les réalise avec un binôme ;
- Lorsque tous les tests fonctionnels passent, le produit est livré.

Les méthodes traditionnelles et les méthodes agiles diffèrent de leur approche de fonctionnement. La comparaison repose sur un grand ensemble de critères d'appréciation.

III Etude Comparative entre les méthodes traditionnelles et agiles

Dans cette partie, nous allons détailler les différences existant entre les deux méthodes par un l'aide d'un tableau récapitulatif.

Tableau 2: ***Tableau de comparaison entre la méthode traditionnelle et agile***

Thèmes	Approche traditionnelle	Approche agile
Cycle de vie	En cascade ou en V, sans rétroaction possible, phases séquentielles	Itérative et incrémentale
Planification	Prédictive, caractérisée par des plans plus ou moins détaillés sur la base d'un périmètre et d'exigences définies et stables au début	Adaptive avec plusieurs niveaux de planification (macro et micro planification) avec ajustements si nécessaires au fil de l'eau en fonction des changements survenus
Documentation	Produite en quantité importante comme support de communication, de validation et de contractualisation	Réduite au strict nécessaire au profil d'incréments fonctionnels opérationnels pour obtenir le feedback du client
Equipe	Une équipe avec des ressources spécialisées, dirigées par un chef de projet	Une équipe responsabilisée où l'initiative et la communication sont privilégiées, soutenue par le chef de projet
Qualité	Contrôle qualité à la fin du cycle de développement. Le client découvre le produit fini.	Un contrôle qualité précoce et permanent, au niveau du produit et du processus. Le client visualise les résultats tôt et fréquemment

Changement	Résistance voire opposition au changement. Processus lourds de gestion des changements acceptés.	Accueil favorable au changement inéluctable, intégré dans le processus.
Suivi de l'avancement	Mesure de la conformité aux plans initiaux. Analyse des écarts	Un seul indicateur d'avancement : le nombre de fonctionnalités implémentées et le travail restant à faire.
Gestion des risques	Processus distinct, rigoureux, de gestion des risques.	Gestion des risques intégrée dans le processus global, avec responsabilisation de chacun dans l'identification et la résolution des risques. Pilotage par les risques.
Mesure du succès	Respect des enseignements initiaux en termes de coûts, de budget et de niveau de qualité	Satisfaction client par la livraison de valeur ajoutée

Avec toutes ces différences énumérées, nous pouvons constater que les méthodes agiles semblent être plus adéquates pour des projets itératifs et incrémentaux. Avec l'approche agile, le client est donc plus impliqué et il peut à chaque fois, exprimer de nouveaux besoins à prendre en compte sur le projet. Parmi toutes les méthodes agiles existant, notre choix s'est orienté sur SCRUM, car son utilisation est plus simple et plus adaptée à notre projet.

IV La Présentation du Framework SCRUM

Scrum est sans doute la méthode agile de gestion de projet la plus répandue et la plus utilisée à l'heure actuelle, reléguant la méthode en cascade au rang d'ancêtre. Le déroulement d'un projet Scrum diffère grandement de celui très classique du cycle en V, et nécessite généralement un changement d'état d'esprit de la part des équipes qui ne sont pas habituées à travailler de façon agile. Voyons donc ce qui constitue le cycle projet Scrum.

Il faut noter que trois rôles principaux sont définis dans le processus Scrum :

- Le Scrum Master a pour rôle de vérifier l'application des principes de Scrum, s'assurer que la communication fonctionne correctement au sein de l'équipe et explorer toutes les pistes d'améliorations en termes de productivité et de savoir-faire ;
- L'équipe Scrum regroupe généralement une dizaine de personnes (mais de très gros projets peuvent rassembler plusieurs dizaines de personnes) chargées de la réalisation des différentes tâches. Chaque membre est au même niveau, il n'y a pas de hiérarchie et chacun peut contribuer en fonction de ses compétences ;
- Enfin, le Product Owner est un expert du processus métier du client ; il définit les spécifications fonctionnelles et arbitre les priorités dans les réalisations **[2].**

V. Les cycles du Framework SCRUM

La description des fonctionnalités du produit se fait dans des **Users Stories**. Chaque User Story comprend un identifiant, un nom (descriptif court), une priorité, une estimation de la charge de travail, la description d'un test de validation et éventuellement des commentaires. Une User Story est donc un élément des spécifications fonctionnelles.

Un référentiel des besoins est créé, le **product backlog**. Il va regrouper l'ensemble des Users Stories décrites, donc l'ensemble des fonctionnalités désirées, hiérarchisées par ordre de réalisation souhaitée. Toutes les fonctionnalités ne seront pas nécessairement placées dans le backlog dès le début du projet. La product backlog va évoluer durant tout le projet à partir des nouvelles fonctionnalités voulues par le client.

Le déroulement du cycle projet scrum est ensuite organisé en itérations, ou sprints. Chaque sprint dure généralement de deux à six semaines, rarement plus. Une réunion de planification est organisée avant le début d'un sprint afin de sélectionner les Users Stories qui seront réalisées durant le sprint. Elles composent alors le sprint backlog.

Lors du déroulement d'un sprint, une réunion quotidienne d'avancement est organisée avec l'intégralité de l'équipe. C'est l'occasion de faire le point sur le déroulement du sprint et de vérifier que les objectifs définis seront tenus. Chaque membre de l'équipe peut faire part de son avancement, mais également des difficultés qu'il rencontre et recevoir éventuellement l'aide ou simplement des réponses des autres participants. Cette réunion est très importante, car elle permet à tous les membres de l'équipe d'avoir le même niveau d'information et au scrum master de suivre le déroulement du projet. L'image ci-dessous décrit clairement le fonctionnement du Framework SCRUM.

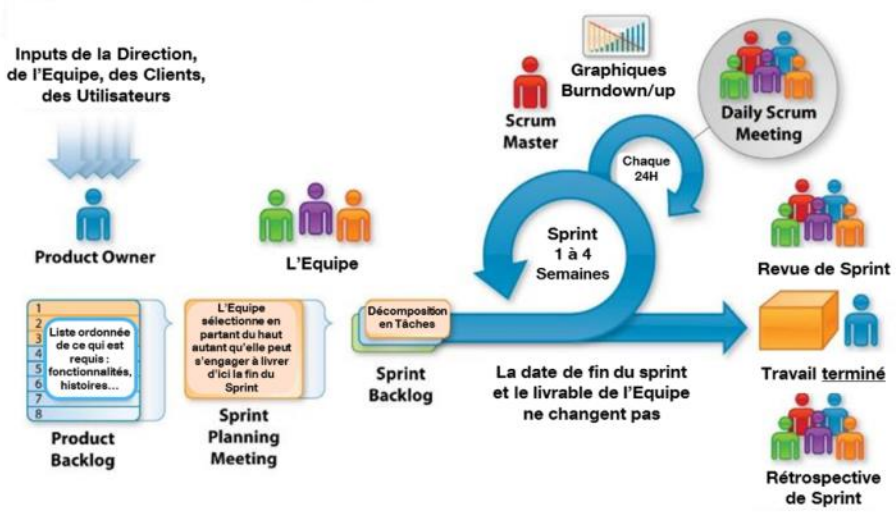


Figure 3: ***Procédé de Fonctionnement du Framework SCRUM***

Contrairement à une méthode classique comme la cascade, le cycle projet scrum permet une plus grande adaptabilité de l'équipe. Le point quotidien permet de repérer rapidement toute difficulté et ainsi prévenir tout dérapage dans le déroulement du projet. L'usage du product backlog, des sprints et des démonstrations apportent à la fois la souplesse et la rigueur nécessaire à la réalisation du projet et à la satisfaction du client **[2].**

VI. Organisation de l'équipe avec le Framework SCRUM

1. L'organisation de l'équipe projet

Nous allons définir l'organisation de l'équipe projet selon le tableau défini ci-dessous. L'organisation du projet s'appuie sur la méthode SCRUM.

Tableau 3: ***Organisation de l'équipe projet SCRUM***

Personnes	Rôles	Fonctions
Mr Alain Charles GOMIS	Product Owner	Directeur des Ressources Humaines de l'UASZ
Mr. Alioune Badara DIENG	Product Owner	Chef des Services Pédagogiques de l'UFR des Science et Technologies de l'UASZ

Dr. Khadim DRAME	Scrum Master	Enseignant chercheur au département d'Informatique de l'UASZ
M. Adama SEYE	Scrum Team	Etudiant en master 2 Informatique à l'UASZ

2. ***Déroulement du projet sur SCRUM***

Nous avons donc défini l'organisation de notre travail selon la méthodologie SCRUM. Chaque membre de l'équipe joue pleinement son rôle et participe à l'élaboration du projet.

Nous avons fixé des entretiens avec nos Product Owner. Ces entretiens n'étaient pas faciles à organiser, car à chaque fois, on repoussait pour une date ultérieure. Cela coïncidait toujours à un empêchement de dernière minute. Nous avons pu faire nos entretiens et avons bien défini les besoins fonctionnels attendus par les clients.

Chaque quinzaine, on organise avec le scrum master une réunion sur son bureau pour faire l'état des lieux sur l'avancement du travail et il définit de nouveaux objectifs à atteindre. Malgré les contraintes de temps, il est prévu de faire une rencontre avec le scrum master, Product Owner et le scrum Team pour revoir l'avancement du projet.

Les méthodes agiles impliquent d'avantage l'attention des clients. A chaque fois que de nouveaux besoins se font exprimés, on essaiera de les inclure sur le projet en cours de développement. Le processus de développement est purement incrémental.

Les méthodes agiles nous permettent donc une bonne planification et une organisation par étapes courtes sur l'incrémentation du projet.

Le chapitre suivant est une illustration de l'utilisation du Framework scrum dans la spécification et l'analyse des besoins exprimés par les clients.

CHAPITRE III : SPECIFICATION ET ANALYSE DES BESOINS FONCTIONNELS

Dans ce chapitre, il sera question d'analyser l'ensemble des besoins exprimés par les clients et de proposer des solutions en mettant en place des modèles d'analyse nécessaires pour la réalisation de notre application.

Notre travail s'appuiera essentiellement sur le langage UML (Unified Modeling Language) dont la traduction française est « langage de modélisation unifié). C'est un langage de modélisation purement orienté objet.

UML nous permet de mettre en place des modèles spécifiques qui permettent de cerner le champ d'étude d'un projet. Il décrit clairement les besoins spécifiques du projet et propose des solutions fonctionnelles et techniques grâce à l'utilisation de ses diagrammes.

Actuellement, UML comporte quatorze types de diagramme (diagramme de profils qui est le nouveau), représentant autant de vue distincte pour représenter des concepts particuliers du futur système. La modélisation s'effectue sous deux principaux aspects que sont la modélisation statique (vue statique du système) et la modélisation dynamique (vue dynamique avec les interactions entre objets).

Nous traduirons dans cette partie, les besoins exprimés en spécifications fonctionnelles grâce à l'utilisation d'UML. Et après, nous essayerons de faire une analyse des besoins en proposant des modèles.

I. Spécification des besoins

Dans cette partie, il est nécessaire de délimiter le contexte du sujet en faisant une étude fonctionnelle du futur système. Il s'agit donc de montrer l'ensemble des acteurs agissant au sein de notre système et d'identifier les cas d'utilisation spécifiques.

1. Identification des acteurs

Un acteur est l'archétype de l'utilisateur (personne, processus externe, …) qui interagit avec le système. Par défaut, c'est un acteur principal, c'est-à-dire qu'il agit directement sur le système et en attend des résultats ou bien, il peut être un acteur secondaire qui est souvent sollicité pour des informations supplémentaires.

Dans notre cas, le tableau ci-dessous détaille l'ensemble de nos acteurs agissant sur le système.

Tableau 4: ***Identification des acteurs***

Acteurs	Descriptions	Rôles
Administrateur	Il peut être le DRH ou tout autre personne qualifiée	Il gère la partie administration de la plateforme
Superviseur	Le superviseur fait partie du PATS, c'est soit le CSP, le DRH, le CSA, etc. en fonction du type d'élections.	Il supervise le déroulement de l'élection ; il fait l'ouverture ainsi que la fermeture du dépôt des candidatures ; il publie la liste des candidats, et le PV des résultats de vote ; il peut paramétrer une élection mais aussi envoyer des notifications concernant une élection donnée.
Candidat	Pour chaque type d'élections, on a un type spécifique de candidat (étudiant, PER, ou PATS).	Il peut modifier ses informations ou retirer sa candidature.
Electeur	Les électeurs peuvent être membre du PATS, PER ou des étudiants	Il peut déposer sa candidature pour une élection donnée ; il peut aussi voter pour le candidat de son choix.

2. ***Identification des fonctionnalités***

Les fonctionnalités sont les actions ou services pouvant être rendus par le système en cas de sollicitation de l'acteur concerné. Ces services peuvent être externes (nécessitant un acteur) ou bien, interne, c'est-à-dire non-visible, mais fonctionnant en interne s'il y a le démarrage d'un processus donné. Dans notre étude, nous avons pu lister un ensemble de fonctionnalités pouvant mener à bien l'organisation et le processus de vote dans notre système. Le tableau ci-dessous montre l'ensemble des fonctionnalités.

Tableau 5: ***Identification des fonctionnalités***

Fonctionnalités	Acteur(s)
S'authentifier	Admin, Candidat, Electeur, Superviseur
Créer une élection	Admin
Gérer les utilisateurs (ajouter, modifier, supprimer)	Admin
Gérer les postes liés à une élection	Admin
Attribuer une élection à un superviseur	Admin
Gérer les UFR	Admin
Gérer les Départements	Admin
Gérer les services	Admin
Publier note d'information sur l'ouverture du scrutin par courriel électronique	Superviseur
Voir la liste des candidats inscrits pour une élection	Superviseur
Voir la liste des électeurs d'une élection	Superviseur
Visualiser l'avancé du vote (visualiser électeurs votant et électeurs non votant) et les résultats provisoires	Superviseur
Publier les résultats avec impression du procès-verbal	Superviseur
Déposer sa candidature	Electeur
Modifier ses informations	Candidat
Retirer sa candidature	Candidat
Effectuer un vote	Electeur, Candidat
Ouvrir une élection	Superviseur
Valider une candidature	Superviseur
Retirer un candidat	Superviseur

3. ***Diagramme de cas d'utilisation***

Le diagramme de cas d'utilisation est une représentation des principales fonctionnalités nécessaires à l'étude de notre système. Il décrit le système étudié en privilégiant le point de vu utilisateur.

Il permet ainsi de recueillir, d'analyser et d'organiser les besoins, et de recenser les principales fonctionnalités d'un système. Il montre aussi les interactions existant entre acteurs et cas d'utilisation, entre cas d'utilisation (inclusion, extension, généralisation), et entre acteurs (généralisation). Dans notre cas, nous avons trois diagrammes de cas d'utilisation selon les acteurs concernés.

a) *Diagramme de cas d'utilisation de l'administrateur*

Le diagramme de cas d'utilisation de l'administrateur permet de représenter les fonctionnalités auxquelles cet acteur a accès à notre système.

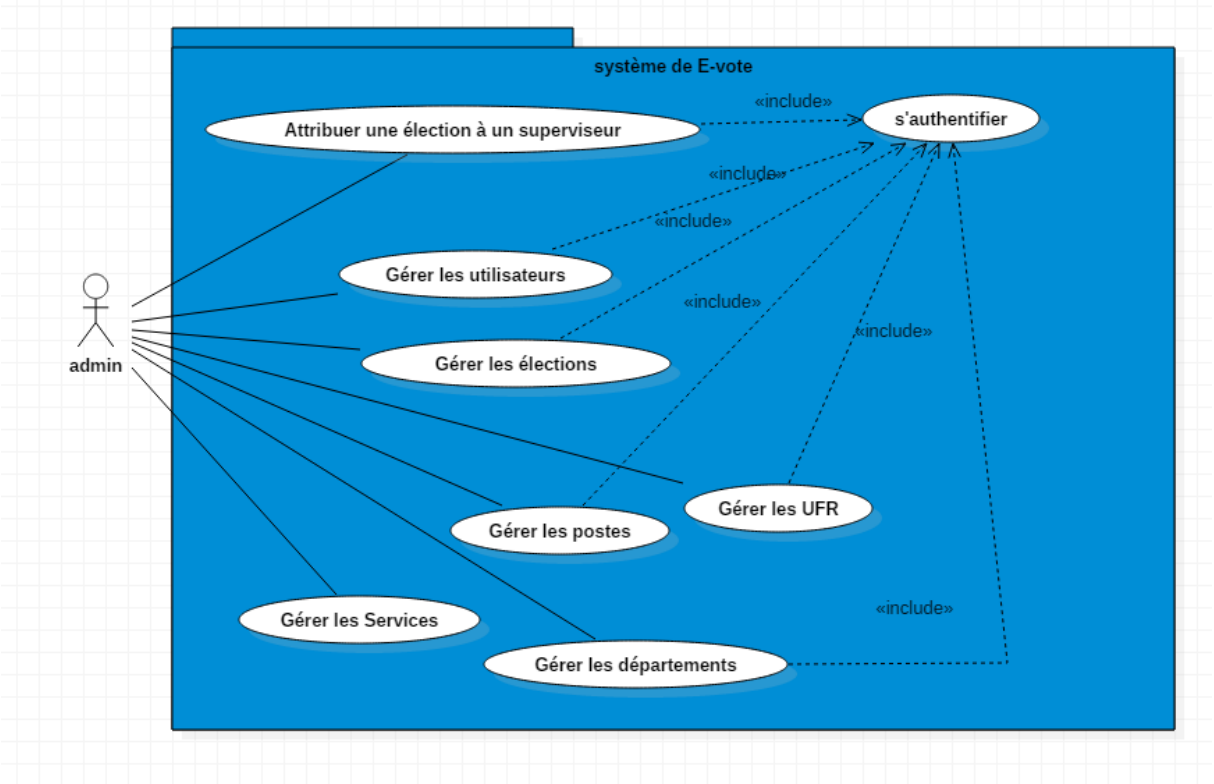


Figure 4: ***Diagramme de cas d'utilisation de l'administrateur***

Dans le diagramme, ci-dessus, nous voyons que l'administrateur a un grand nombre de fonctionnalités à gérer ; et pour chaque fonctionnalité, il a besoin de s'authentifier. S'il y a de nouveaux membres, il doit les enregistrer au niveau de la base de données de l'application. Il créé et attribue une élection à un superviseur.

b) *Diagramme de cas d'utilisation du superviseur*

Le diagramme suivant décrit toutes les fonctionnalités attribuées au superviseur :

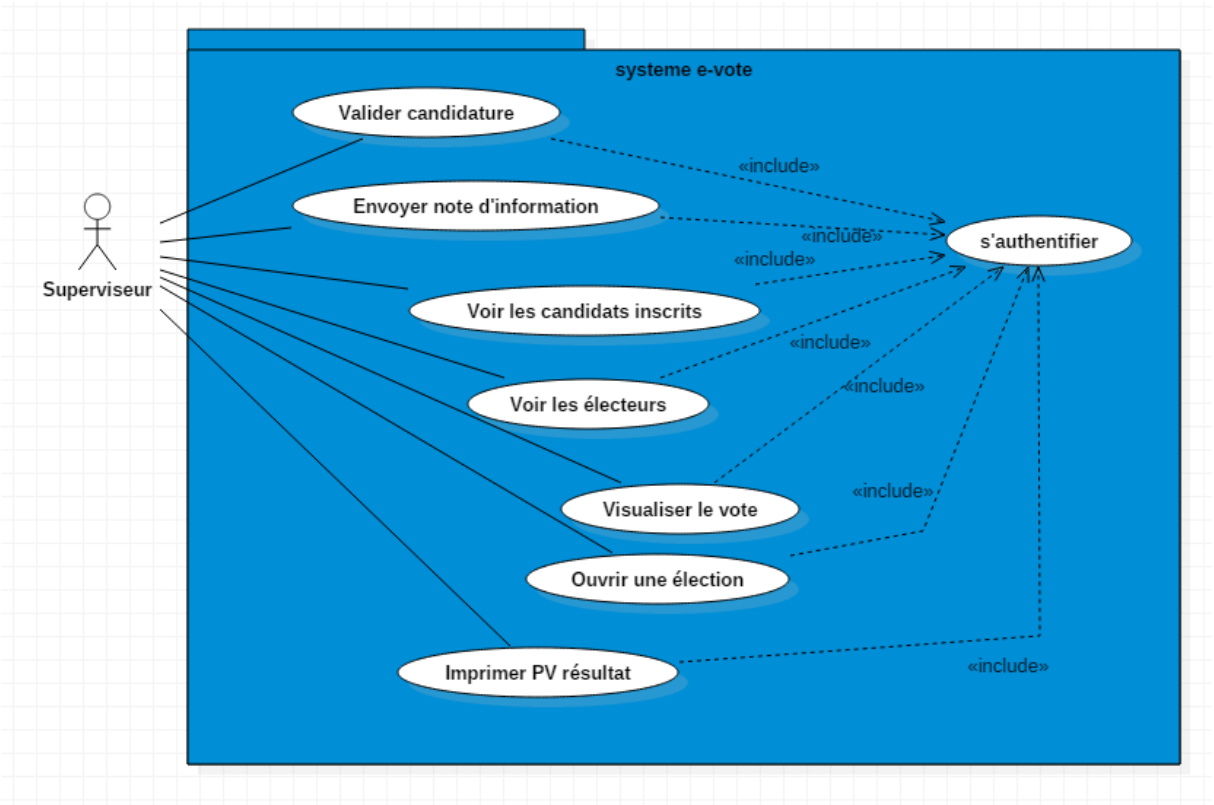


*Figure 5**:Diagramme de cas d'utilisation du superviseur***

Le superviseur gère toute la procédure de l'élection, il envoie une note d'information aux électeurs dès l'ouverture de l'élection. Il valide ainsi les candidatures déposées et envoie en retour la liste des candidats inscrits. Au déroulement du scrutin, il peut visualiser les résultats provisoires ainsi que les taux de participation et publier le PV de résultat.

c) *Diagramme de cas d'utilisation des électeurs et candidats*

Le diagramme ci-dessous montre quelques fonctionnalités faites par les électeurs et les candidats :

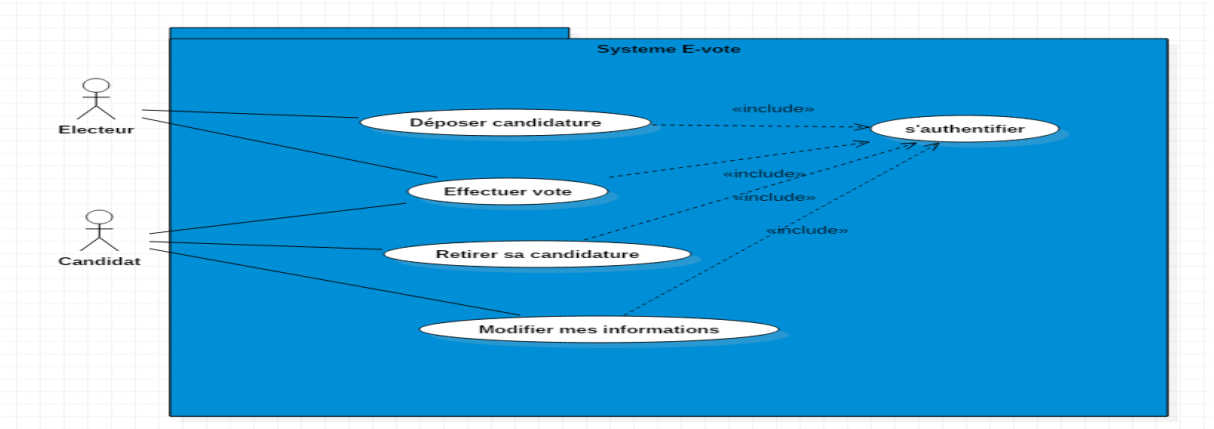


Figure 6: ***Diagramme de cas d'utilisation électeur et candidat***

Dans ce diagramme, nous voyons que l'électeur peut déposer sa candidature. Cette fonctionnalité est activée lorsqu'il n'y a pas encore déroulement du scrutin. Tous les utilisateurs du système sont d'abord considérés comme des électeurs. L'électeur et le candidat peuvent voter. Le candidat quant à lui peut à chaque fois modifier ses informations ou même retirer sa candidature.

4. *Description des cas d'utilisation*

Dans cette partie, nous allons faire la description de quelques cas d'utilisation de notre diagramme. La description des cas se fera sous forme de tableau.

a. *Description du cas d'utilisation « s'authentifier »*

Tableau 6: ***Description du cas d'utilisation "s'authentifier"***

Identification	
Cas numéro	1
Nom	S'authentifier
Acteur(s)	tous les utilisateurs du système
Description	Avant toute manipulation du système, faudra procéder à une authentification
Auteur	Adama SEYE
Date(s)	20/06/2018
Préconditions	Aucun
Démarrage	l'utilisateur lance l'application et essaie d'entrer dans une fonctionnalité
Scénario nominal	
1	Au lancement de l'application, un formulaire d'authentification est lancé
2	Dans ce formulaire, il demande à l'utilisateur de donner son login et son mot de passe
3	Si les informations saisies par l'utilisateur sont correctes, il lui envoie un message confirmant connexion réussie et on note une ouverture du formulaire voulu
Scénarios alternatifs	
1a	L'utilisateur peut décider de quitter même l'authentification
2a	L'utilisateur peut entrer un login et un mot de passe incorrect où il devra réessayer la saisie
3a	L'accès sera refusé à toute personne ne faisant pas partie de la base (qui n'est ni un électeur, ni un candidat, ni un superviseur)
Post condition (aucun)	
Fin (fin à l'étape 3, 1a ,2a, 3a)	

b. *Description du cas d'utilisation « voir liste des candidats »*

Tableau 7: ***Description du cas d'utilisation "voir liste des candidats"***

Identification	
Cas numéro	2
Nom	Voir liste des candidats
Acteur(s)	Superviseur
Description	Il permet de lister l'ensemble des candidats pour chaque élection
Auteur	Adama SEYE
Date(s)	20/06/2018
Préconditions	Il faudra s'authentifier
Démarrage	L'utilisateur sera redirigé vers la liste des candidats
Scénario nominal	
1	Le menu superviseur est lancé avec les différentes options
2	Il peut ainsi aller vers l'option voir liste candidats
Scénarios alternatifs	
2a	L'utilisateur peut décider de quitter même le menu d'accueil
Post condition (aucun)	
Fin (fin à l'étape 2a)	

c. *Description du cas d'utilisation « voir liste des électeurs »*

Tableau 8: ***Description du cas d'utilisation "voir liste des électeurs"***

Identification	
Cas numéro	3
Nom	Voir liste des électeurs
Acteur(s)	Superviseur
Description	Il permet de voir l'ensemble des électeurs d'une élection donnée
Auteur	Adama SEYE
Date(s)	20/06/2018
Préconditions	Il faudra s'authentifier
Démarrage	L'utilisateur sera redirigé vers le menu superviseur
Scénario nominal	
1	L'authentification réussie, l'utilisateur sera redirigé vers le menu superviseur
2	Il peut ainsi aller vers l'option voir les électeurs
Scénarios alternatifs	
1a	En cas d'authentification invalide, l'utilisateur n'accèdera au système
2a	Il peut décider de quitter le menu en se déconnectant
Post condition (aucun)	

Fin (fin à l'étape 1a ou 2a)

d. *Description du cas d'utilisation « visualiser le vote »*

Tableau 9: ***Description du cas d'utilisation "visualiser le vote"***

Identification	
Cas numéro	4
Nom	Visualiser le vote
Acteur(s)	Superviseur
Description	Il permet de voir l'évolution du vote et de voir ceux qui ont voté et ceux qui n'ont pas encore voté
Auteur	Adama SEYE
Date(s)	20/06/2018
Préconditions	Il faudra s'authentifier
Démarrage	L'utilisateur sera redirigé vers le menu superviseur
Scénario nominal	
1	L'authentification réussie, l'utilisateur sera redirigé vers le menu superviseur
2	Il peut ainsi aller vers l'option visualiser l'avancée du vote
Scénarios alternatifs	
1a	En cas d'authentification invalide, l'utilisateur n'accèdera au système
2a	Il peut décider de quitter le menu en se déconnectant
Post condition (aucun)	
Fin (fin à l'étape 1a ou 2a)	

e. *Description du cas d'utilisation « Envoyer note d'information »*

Tableau 10: ***Description du cas d'utilisation "envoyer note d'information"***

Identification	
Cas numéro	5
Nom	Envoyer note d'information
Acteur(s)	Superviseur
Description	Il permet d'envoyer des notifications pour avertir les électeurs sur l'ouverture du scrutin
Auteur	Adama SEYE
Date(s)	20/06/2018
Préconditions	Il faudra s'authentifier
Démarrage	L'utilisateur sera redirigé vers le menu superviseur
Scénario nominal	
1	L'authentification réussie, l'utilisateur sera redirigé vers le menu superviseur

2	Il peut ainsi aller vers l'option envoyer note d'information
3	Une fois vers cette page, il peut cibler les destinataires et les envoyer la note d'information
4	L'information sera reçue sur les boites mail des étudiants ou bien au niveau de la plateforme
Scénarios alternatifs	
1a	En cas d'authentification invalide, l'utilisateur n'accèdera au système
2a	Il peut décider de quitter le menu en se déconnectant
3a	Il peut à partir de la page annuler l'envoi en se déconnectant
Post condition (aucun)	
Fin (fin à l'étape 1a ou 2a ou 3a)	

f. Description du cas d'utilisation « valider candidature »

Tableau 11: ***Description du cas d'utilisation "valider candidature"***

Identification	
Cas numéro	6
Nom	Valider candidature
Acteur(s)	Superviseur
Description	Il permet au superviseur de vérifier si les critères d'éligibilité sont remplies ou pas
Auteur	Adama SEYE
Date(s)	20/06/2018
Préconditions	Il faudra s'authentifier
Démarrage	L'utilisateur sera redirigé vers le menu superviseur
Scénario nominal	
1	L'authentification réussie, l'utilisateur sera redirigé vers le menu superviseur
2	Il peut ainsi aller vers la liste des candidats
3	Une fois vers la liste des candidats, il peut consulter les informations de chaque candidats (sa filière, son statut, son niveau, etc.)
4	Si un candidat ne remplit pas les critères, il peut être retiré de la liste des candidats potentiels
Scénarios alternatifs	
1a	En cas d'authentification invalide, l'utilisateur n'accèdera au système
2a	Il peut décider de quitter le menu en se déconnectant
3a	Il peut quitter la page des candidats en se déconnectant
Post condition (aucun)	
Fin (fin à l'étape 1a ou 2a ou 3a)	

g. Description du cas « imprimer procès-verbal résultat »

Tableau 12: ***Description du cas d'utilisation "imprimer procès-verbal des résultats"***

Identification	
Cas numéro	7
Nom	Imprimer procès-verbal des résultats
Acteur(s)	Superviseur
Description	Il permet d'imprimer le résultat final des votes
Auteur	Adama SEYE
Date(s)	20/06/2018
Préconditions	Il faudra s'authentifier, Il faudra que le vote soit terminé (le système fera le comptage automatique des voix obtenus pour chaque candidat)
Démarrage	L'utilisateur sera redirigé vers le menu superviseur
Scénario nominal	
1	L'authentification réussie, l'utilisateur sera redirigé vers le menu superviseur
2	Il peut ainsi aller vers la page des résultats de vote en imprimant le résultat
3	Une fois vers la liste des candidats, il peut consulter les informations de chaque candidats (sa filière, son statut, son niveau, etc.)
Scénarios alternatifs	
1a	En cas d'authentification invalide, l'utilisateur n'accèdera au système
2a	Il peut décider de quitter le menu en se déconnectant
3a	Il peut quitter la page des candidats en se déconnectant
Post condition (aucun)	
Fin (fin à l'étape 1a ou 2a ou 3a)	

h. *Description du cas d'utilisation « Déposer candidature »*

Tableau 13: ***Description du cas d'utilisation "Déposer candidature"***

Identification	
Cas numéro	9
Nom	Déposer Candidature
Acteur(s)	Electeur
Description	Il permet de déposer sa candidature pour un poste donné
Auteur	Adama SEYE
Date(s)	20/06/2018
Préconditions	Il faudra s'authentifier,
Démarrage	L'utilisateur sera redirigé vers le menu utilisateur (il n'a pas le rôle de User)
Scénario nominal	
1	L'authentification réussie, l'utilisateur sera redirigé vers le menu utilisateur
2	Il peut donc aller vers l'option « déposer candidature »
3	Dans la page de dépôt, il va renseigner ses informations et l'élection concernée avec le poste ciblé
Scénarios alternatifs	

1a	En cas d'authentification invalide, l'utilisateur n'accèdera pas au système
2a	Il peut décider de quitter son menu en se déconnectant
3a	Il peut quitter la page de dépôt de candidature
4a	Si des informations erronées sont envoyées, le dépôt de candidature sera invalide
Post condition (aucun)	
Fin (fin à l'étape 1a ou 2a ou 3a ou 4a)	

i. *Description du cas d'utilisation « effectuer vote »*

Tableau 14: ***Description du cas d'utilisation "effectuer vote"***

Identification	
Cas numéro	11
Nom	Effectuer vote
Acteur(s)	Electeur, Candidat
Description	Il peut effectuer son vote en faisant son choix pour un candidat donné
Auteur	Adama SEYE
Date(s)	20/06/2018
Préconditions	Il faudra s'authentifier,
Démarrage	L'utilisateur sera redirigé vers le menu utilisateur (il n'a pas le rôle de User)
Scénario nominal	
1	L'authentification réussie, l'utilisateur sera redirigé vers le menu utilisateur
2	Il peut donc aller vers l'option « effectuer vote »
3	Il renseigne des informations qui lui permettront d'être orienté vers les candidats spécifiques
4	Une liste des candidats le concernant sera affichée et il va faire son choix
5	Un message de confirmation lui sera envoyé
6	Redirection vers la page de sortie
Scénarios alternatifs	
1a	En cas d'authentification invalide, l'utilisateur n'accèdera pas au système
2a	Il peut décider de quitter son menu
3a	Il peut quitter l'option de « effectuer candidature »
4a	Si les informations renseignées par l'électeur sont erronées, il aura un message d'alerte
Post condition (aucun)	
Fin (fin à l'étape 6, 1a ou 2a ou 3a ou 4a)	

j. *Description du cas d'utilisation « Paramétrer élection »*

Tableau 15: ***Description du cas d'utilisation "paramétrer élection"***

Identification	
Cas numéro	8
Nom	Paramétrer une élection
Acteur(s)	Superviseur
Description	Il permet l'ajout ou bien l'ouverture d'une élection au niveau de la plateforme et qui sera pris en compte pour l'ouverture du scrutin
Auteur	Adama SEYE
Date(s)	20/06/2018
Préconditions	Il faudra s'authentifier,
Démarrage	L'utilisateur sera redirigé vers le menu superviseur -Aller vers l'option paramétrer élection
Scénario nominal	
1	L'authentification réussie, l'utilisateur sera redirigé vers le menu superviseur
2	Il peut donc aller vers l'option paramétrer élection
3	L'élection sera ajoutée et après une note d'information est envoyée aux électeurs ainsi que pour les dépôts de candidature
Scénarios alternatifs	
1a	En cas d'authentification invalide, l'utilisateur n'accèdera au système
2a	Il peut décider de quitter le menu en se déconnectant
3a	Il peut quitter la page de paramétrage d'une élection et se déconnecter
Post condition (aucun)	
Fin (fin à l'étape 1a ou 2a ou 3a)	

II. Analyse des besoins fonctionnels du système

Dans cette partie, nous allons faire l'analyse des besoins fonctionnels du système. Chaque cas d'utilisation présente tout un ensemble de processus dans son fonctionnement. Nous choisirons l'analyse de trois cas d'utilisation en présentant leurs diagrammes d'activité et de séquence.

1. Etude des activités de « déposer candidature »

Nous allons étudier l'enchaînement des scénarios possibles du cas « déposer sa candidature ». Des processus sont donc déclenchés au démarrage du cas. Le diagramme de séquence et le diagramme d'activité sont les parfaites illustrations de l'étude du cas.

a. Diagramme de séquence du cas « déposer candidature »

L'électeur désirant être candidat, devra donc s'authentifier en saisissant son login (qui est son compte mail de l'Université) et son mot de passe d'accès (fourni par le superviseur lors de

l'envoi de la note d'information). Si tout est correct, il est redirigé vers son menu et il choisit l'option « déposer candidature ». Il devra spécifier s'il est étudiant, enseignant ou PATS et après, un formulaire de candidature lui sera envoyé ; il le renseigne et valide sa candidature. Il sera ensuite redirigé vers une page de confirmation.

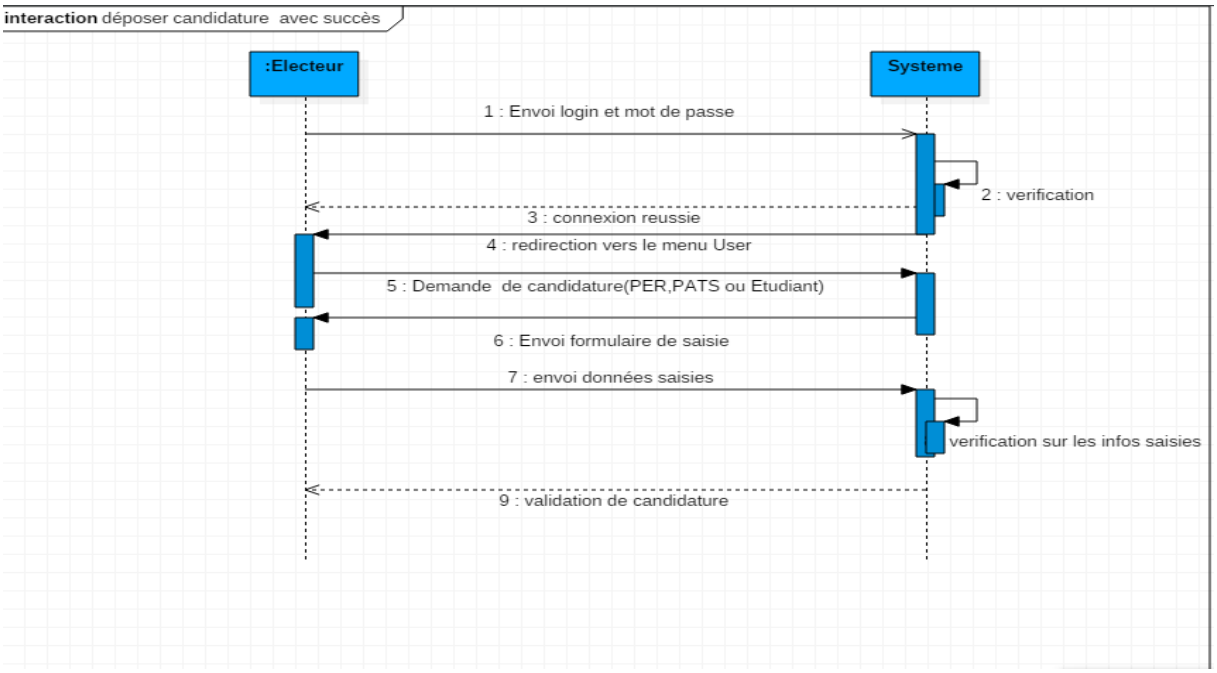


Figure 7: ***Diagramme de séquence "déposer candidature"***

b. *Diagramme d'activité du cas « déposer candidature »*

Dans cette partie, nous retrouvons le même procédé juste que chaque étape déclenche une action spécifique. L'électeur s'authentifie et sera redirigé vers son menu, il choisit l'option « déposer sa candidature » et un formulaire de saisie lui est envoyé.

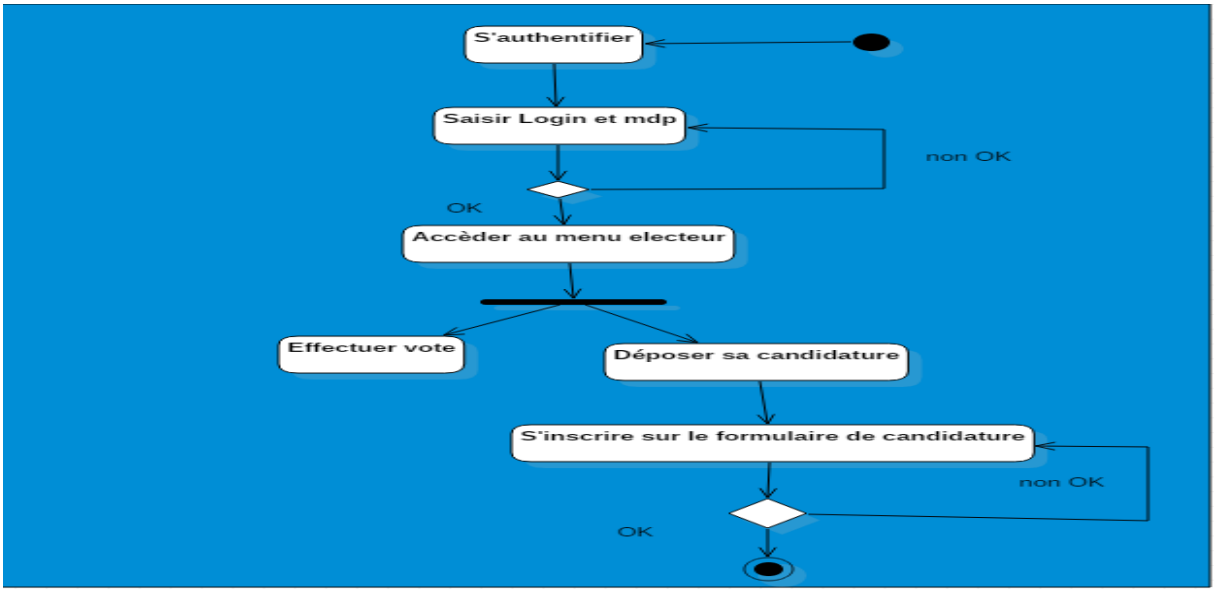


Figure 8: ***Diagramme d'activité de "déposer candidature"***

2. Etude des activités du cas « effectuer vote »

Le cas d'utilisation « effectuer vote » est l'un des cas les plus importants du système, car il permet à l'électeur de pouvoir choisir son candidat.

a. Diagramme de séquence

Pour ce cas, l'électeur connecté, est redirigé vers son menu ; il clique sur l'option « effectuer vote », une liste des élections dont il peut participer lui est présentée. Il choisit une des élections et il accède à la liste des candidats. À ce niveau, une indication lui est donnée sur le nombre de candidat dont il peut voter, il fait ensuite son choix. Si le nombre de candidat dont il peut voter est égal à 0, il est redirigé vers une page de confirmation de vote et il peut retourner vers le menu et faire de même pour les autres élections. Si ce n'est pas le cas, il est redirigé vers une autre page de sortie du scrutin.

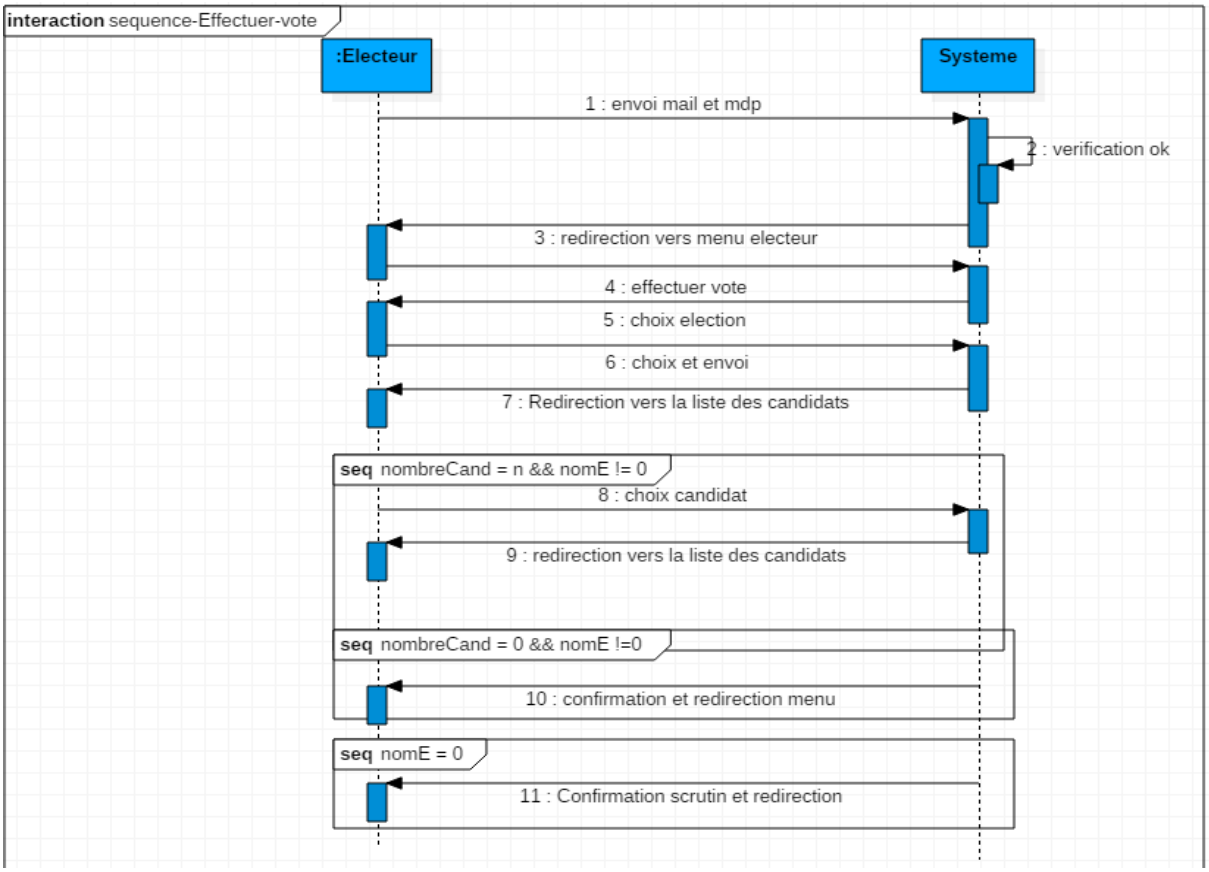


Figure 9: ***Diagramme de séquence "effectuer vote"***

b. Diagramme d'activité

L'utilisateur s'authentifie en saisissant donc son login et son mot de passe d'accès à la plateforme. Si les informations sont correctes, il sera redirigé vers le menu électeur avec

l'option d'effectuer le vote. Il est nécessaire de préciser que le « dépôt des candidatures » et « effectuer vote » ne se font pas en même temps. Un timing est bien défini pour chaque cas. Une vérification interne est faite et il sera redirigé vers la liste des candidats où il pourra faire son choix.

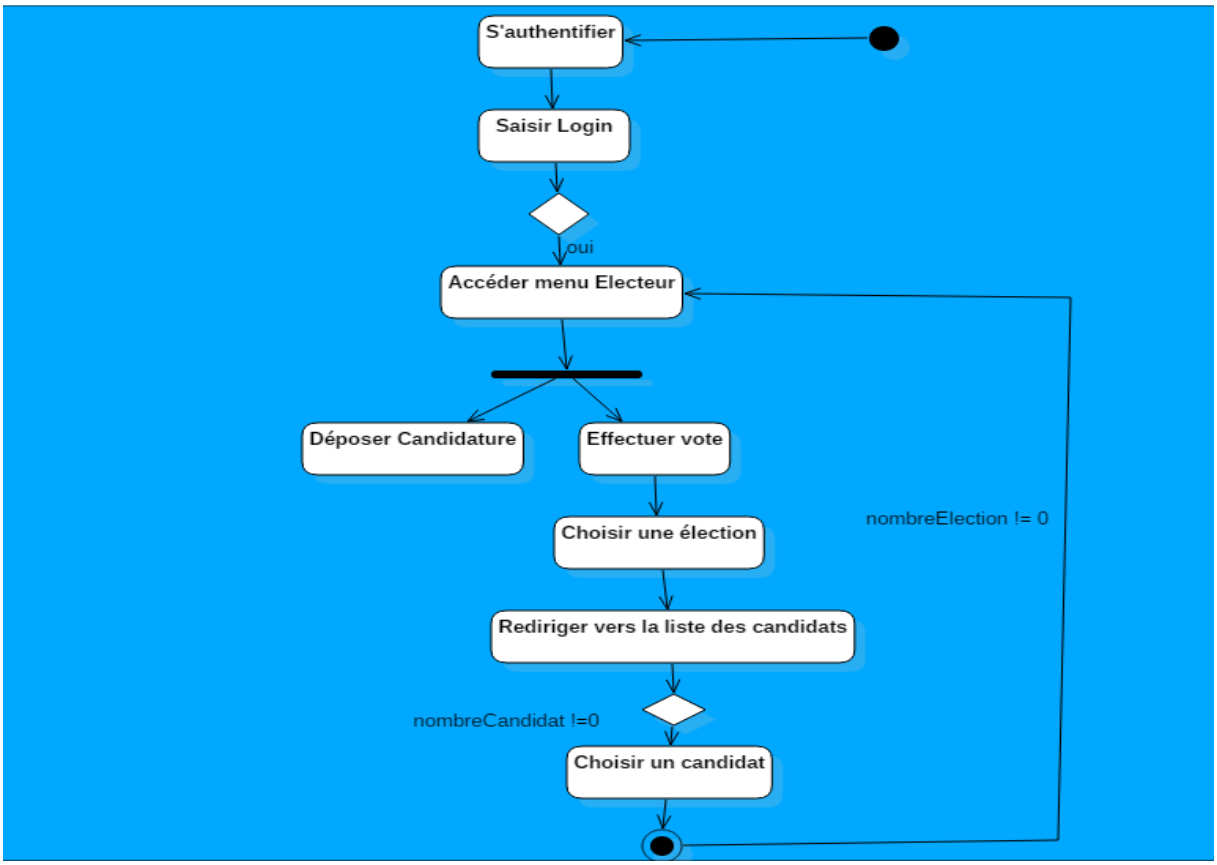


Figure 10: ***Diagramme d'activité "effectuer vote"***

3. *Etude des activités de « visualiser vote »*

Le cas d'utilisation « visualiser le vote » est destiné au superviseur. Il peut donc contrôler l'ensemble du processus de vote. On a donc des statistiques sur le taux de participation à une élection, le nombre de voix pour chaque candidat et à la fin du scrutin, le superviseur pourra imprimer le procès-verbal.

a. *Diagramme de séquence*

Le superviseur s'authentifie en saisissant son login et son mot de passe d'accès. Après une connexion réussie, il est redirigé vers la page correspondant. Il choisit l'option « visualiser le vote », une boîte de dialogue s'ouvre, il choisit sur l'une des élections supervisées, et il est redirigé vers les résultats du scrutin.

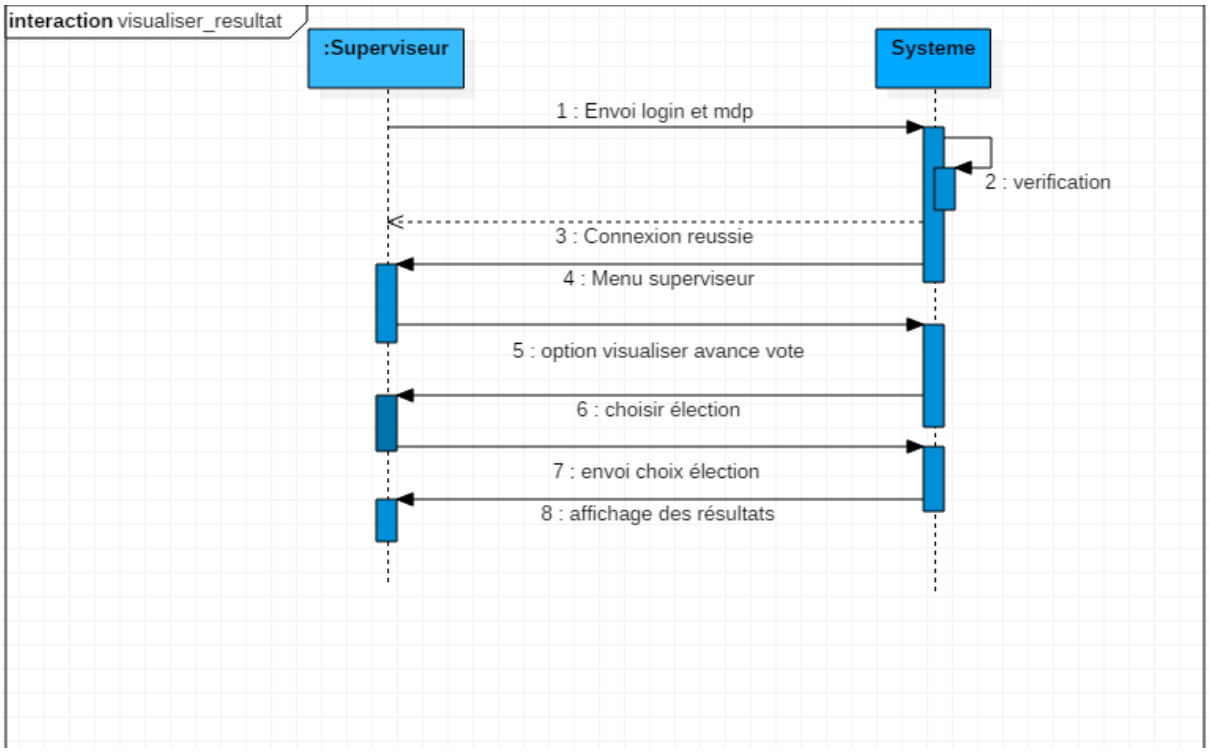


Figure 11: ***Diagramme de séquence "visualiser le vote"***

b. *Diagramme d'activité*

Le superviseur s'authentifie en saisissant son login et son mot de passe d'accès ; si tout est correct, il est redirigé vers le menu superviseur et il peut ainsi choisir l'option « visualiser le vote ».

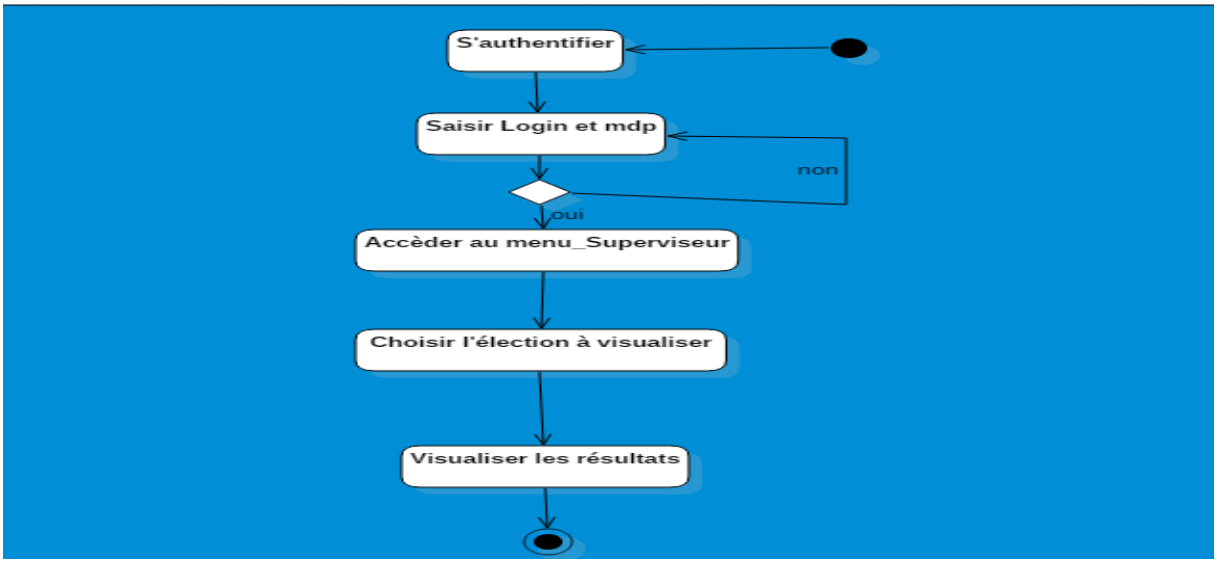


Figure 12: ***Diagramme d'activité du cas "visualiser avancé du vote"***

Ce chapitre a permis de voir en clair l'ensemble des besoins fonctionnels que notre système devra prendre en compte. L'analyse des besoins fonctionnels apporte une grande précision et permet une orientation exacte sur ce qui est attendu. Le chapitre suivant traitera essentiellement la conception du système.

CHAPITRE IV : CONCEPTION DU SYSTEME

La phase de conception est une phase essentielle, car elle permet la concrétisation de notre application en s'appuyant sur des modèles de conception définis sur UML. La conception nécessite une analyse rigoureuse et une proposition de modèle adapté pour la réalisation du système. La conception du système comprend deux étapes que sont la conception générale et la conception détaillée. Dans la conception générale, il faudra définir l'architecture de l'application, le diagramme de packages et le diagramme de déploiement. Pour la conception détaillée, nous allons présenter les différentes classes intervenant et le diagramme de classe.

I. Conception générale du système

Dans cette partie, nous présenterons l'architecture de notre système, le diagramme de packages et le diagramme de déploiement.

1. Architecture du système

Les architectures basées sur le principe « client-serveur » sont les plus utilisées dans le monde du développement Ainsi, nous avons tout un ensemble de modèles d'architecture et le plus utilisé est le modèle MVC. Ce modèle reflète même la présentation de l'architecture 3-Tiers.

a. Modèle MVC

Le pattern MVC (Modèle-Vue-Contrôleur) est l'un des plus célèbres et les plus utilisés dans le développement web. Le pattern MVC permet de bien organiser son code source. Il va vous aider à savoir quels fichiers créés, mais surtout à définir leurs rôles. Le but de MVC est justement de séparer la logique du code en trois parties que l'on retrouve dans des fichiers distincts :

- **Modèle** : cette partie gère les *données*. Son rôle est d'aller récupérer les informations « brutes » dans la base de données, de les organiser et de les assembler pour qu'elles puissent ensuite être traitées par le contrôleur. On y trouve donc entre autres les requêtes SQL.
- **Vue** : cette partie se concentre sur l'*affichage*. Elle ne fait presque aucun calcul et se contente de récupérer des variables pour savoir ce qu'elle doit afficher au niveau du navigateur. On y trouve essentiellement du code HTML, mais aussi quelques boucles et conditions très simples pour afficher par exemple une liste de messages.

- **Contrôleur** : cette partie gère la logique du code qui prend des *décisions*. C'est en quelque sorte l'intermédiaire entre le modèle et la vue : le contrôleur va demander au modèle les données, les analyser, prendre des décisions et renvoyer le texte à afficher à la vue. Le contrôleur contient exclusivement du code métier. C'est notamment lui qui détermine si l'utilisateur a le droit de voir la page ou non (gestion des droits d'accès) **[3].**

La figure ci-dessous décrit le principe de fonctionnement du modèle MVC.

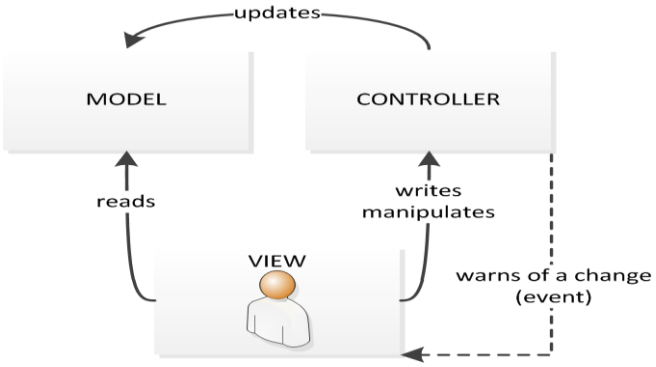


Figure 13: ***Présentation du modèle MVC***

Beaucoup d'architectures s'appuient sur ce modèle. Le plus connu est donc l'architecture 3-Tiers. C'est cette architecture qui sera utilisée pour la réalisation de notre système.

b. Architecture 3-Tiers

L'architecture trois tiers, également appelée architecture à trois niveaux ou à trois couches, est une architecture client-serveur dans laquelle coexistent et sont maintenus des modules indépendants permettant le rendu d'une interface utilisateur (GUI), les process logiques, fonctionnels et métiers ainsi que l'accès aux données. On parle donc ici d'une infrastructure physique qui va servir de support à une infrastructure logicielle (l'infrastructure trois tiers sous-tend l'infrastructure logicielle) **[4].** En effet, n'importe quelle application peut être découpée en trois parties : une partie interface graphique, une partie fonctionnelle, et une partie de stockage de données. Et c'est à ces besoins précis que l'architecture trois-tiers s'est dessinée en découpant trois parties distinctes:

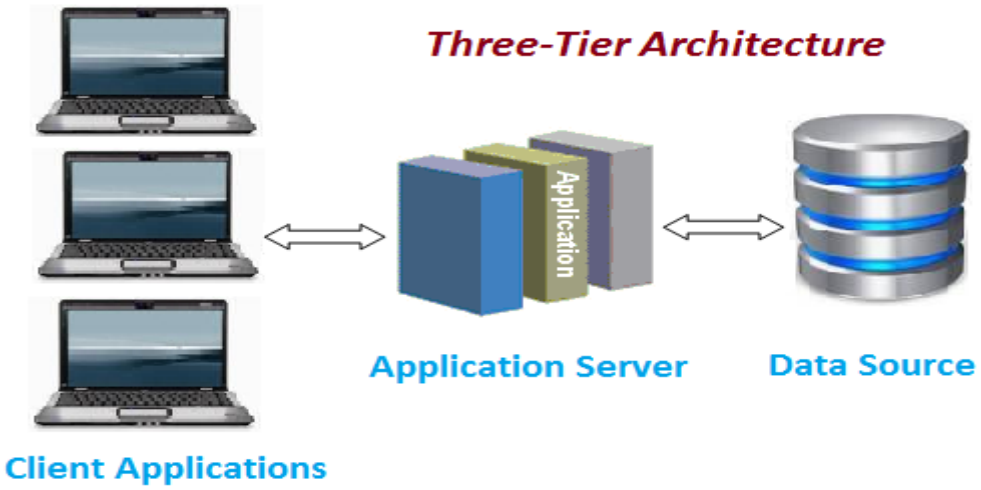


Figure 14: ***Architecture 3 Tiers***

L'architecture 3 tiers nous impose donc à respecter le modèle MVC. L'implémentation de notre système va essentiellement se baser sur ce style architectural.

L'architecture de notre application est représentée par la figure ci-dessous :

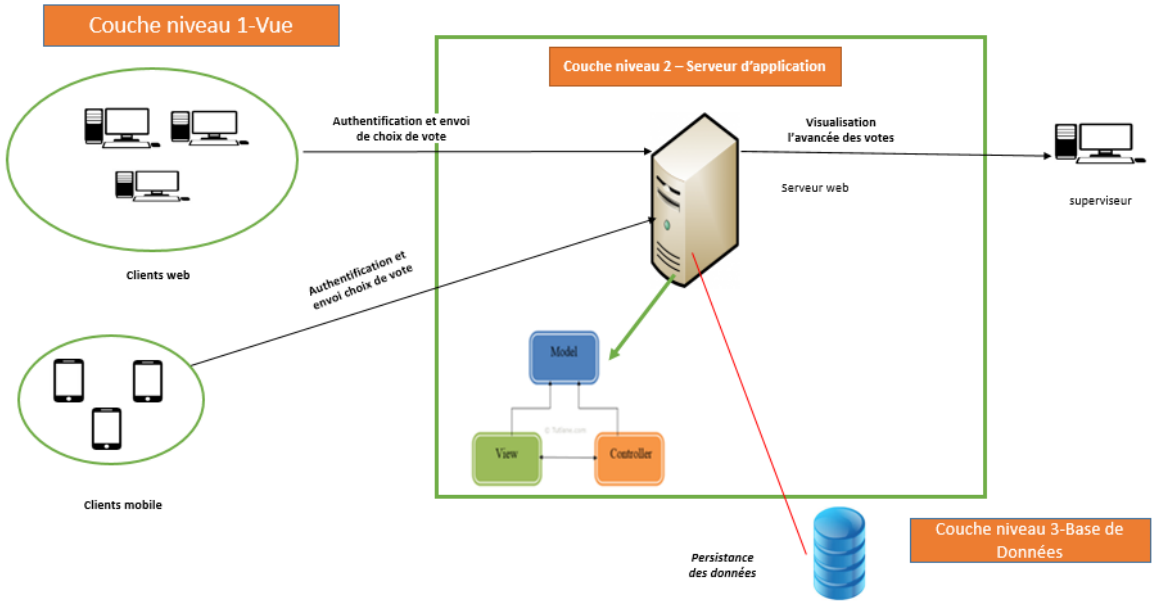


Figure 15: ***L'architecture de notre application***

2. *Diagramme de packages*

Un *diagramme de packages* est un diagramme UML qui fournit une représentation graphique de haut niveau de l'organisation de l'application, et vous aide à identifier les liens de généralisation et de dépendance entre les packages.

Les packages de notre système sont :

- **Authentification** : elle gère l'accès par rapport au profil de l'utilisateur ;
- **Gestion des profils utilisateur** : dans ce package, nous avons les différents menus définis selon le type d'utilisateur. Il est composé de la gestion des menus utilisateurs ;
- **Gestion candidat** : il gère le profil du candidat où il pourra modifier certaines de ses informations ou retirer sa candidature ;
- **Gestion électeur** : il gère le profil de l'électeur où il pourra déposer sa candidature ou voter ;
- **Espace superviseur** : dans ce package, le profil superviseur peut gérer les élections et le suivi ;
- **Gestion Election** : il permet de paramétrer une élection en faisant son ouverture ou ajouter une élection ;
- **Gestion des postes** : il permet d'ajouter un poste lié à une élection donnée ;
- **Suivi résultat** : ce package gère le suivi du scrutin en imprimant le procès-verbal et en envoyant ce PV aux destinataires par mail.

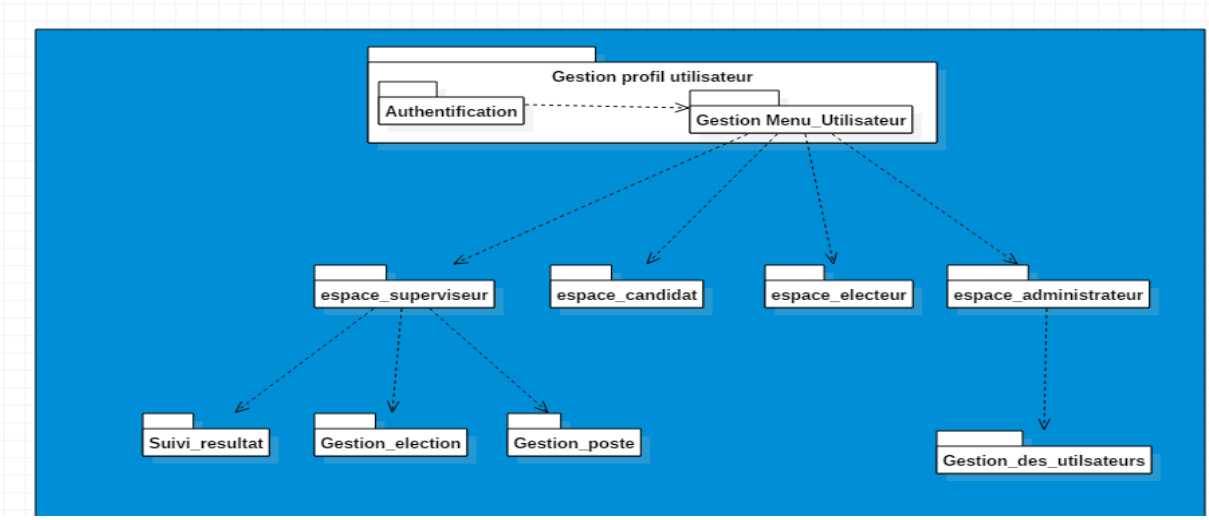


Figure 16: ***Diagramme de Packages***

3. Diagramme de déploiement

En UML, un diagramme de déploiement est une vue statique qui sert à représenter l'utilisation de l'infrastructure physique par le système et la manière dont les composants du système sont répartis ainsi que les relations entre eux.

L'application doit être légère et performante ; elle peut tourner sur des machines de 2Go de RAM et 1.80 GHZ de fréquence processeur. Les ordinateurs tiers pourront accéder au serveur contenu dans une machine d'au moins 8 Go de RAM. La communication est quant à elle assurée par un réseau internet (soit par la technologie Ethernet ou la technologie wifi sécurisé). Le système peut aussi fonctionner en intranet.

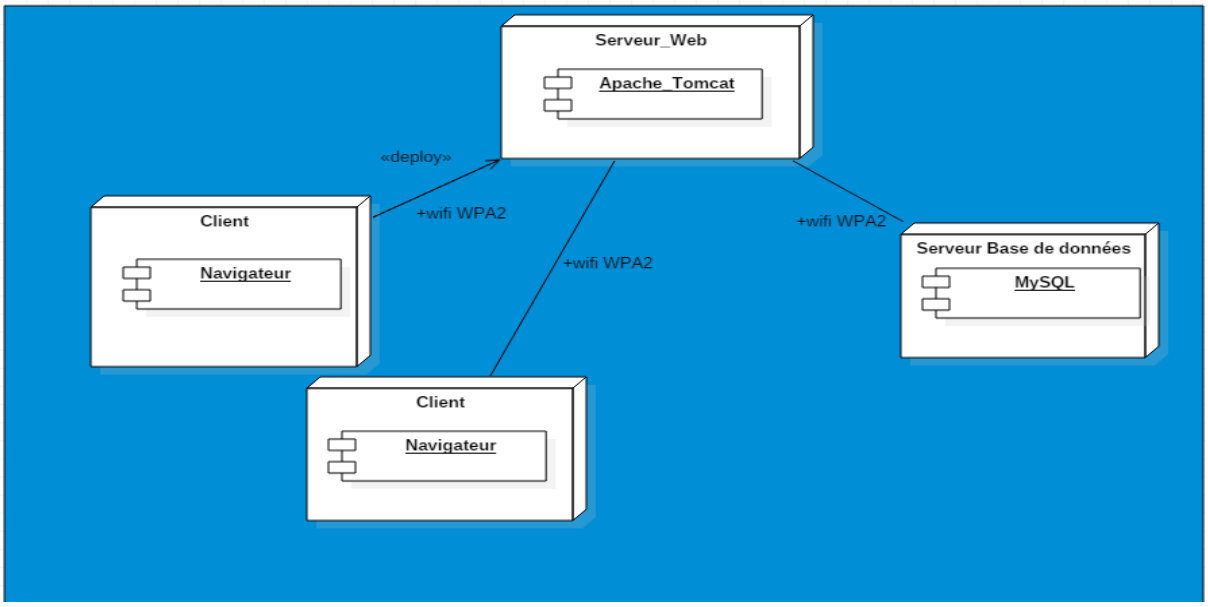


Figure 17: ***Diagramme de déploiement***

II. Conception détaillée du système

La phase de conception détaillée est la dernière phase nous permettant de faire une représentation statique du système à mettre en place. Le diagramme de classes nous permet de représenter l'ensemble des classes participantes à l'implémentation de notre application. La représentation de notre diagramme s'appuie essentiellement sur les concepts de classes associations. Dans cette partie, nous ferons une large description de nos classes et de notre diagramme de classes.

1. Description des différentes classes du système

Dans notre système, nous avons dix-sept classes participantes. Des classes telles que « Electeur », « Superviseur », « Candidat » héritent de la classe « Utilisateur ». Les candidats peuvent être des membres appartenant au PER, PATS ou bien étudiant. Le superviseur gère une élection et les candidats s'inscrivent à une élection. Chaque candidat dispose d'un bulletin de vote qui permettra aux électeurs d'y voter. Chaque élection est liée à un poste brigué. Les électeurs sont considérés comme étant les utilisateurs enregistrés au niveau de la base de données de l'Université, les utilisateurs appartiennent soit à un service, ou à un département qui est lié à une UFR ; en ce qui concerne les étudiants, ils sont dans un département pour un niveau donné (licence1, licence2, etc.). Un superviseur peut envoyer des notifications au niveau de la plateforme qui seront visibles aux électeurs connectés.

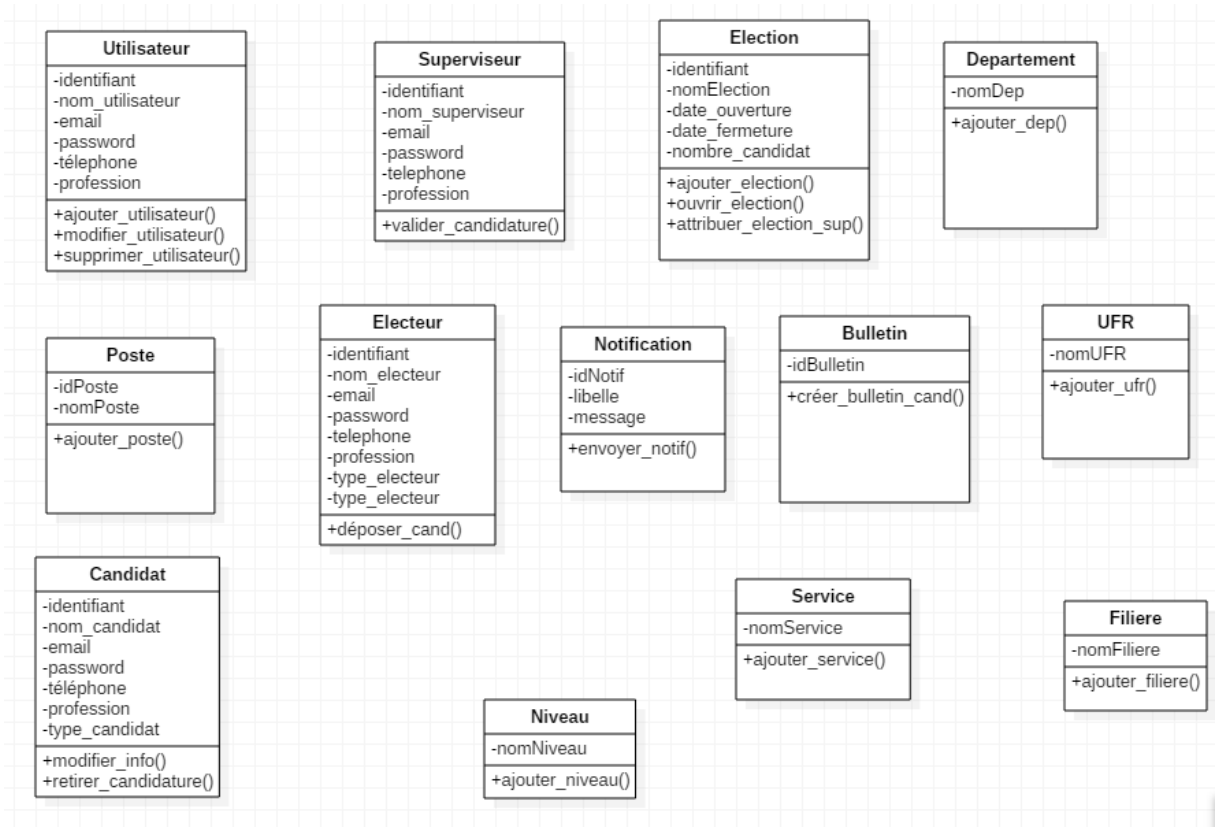


Figure 18:***Les différentes classes du système***

2. Diagramme de classes du système

Dans la figure, ci-dessous, nous allons représenter l'ensemble des classes participantes à l'implémentation du système.

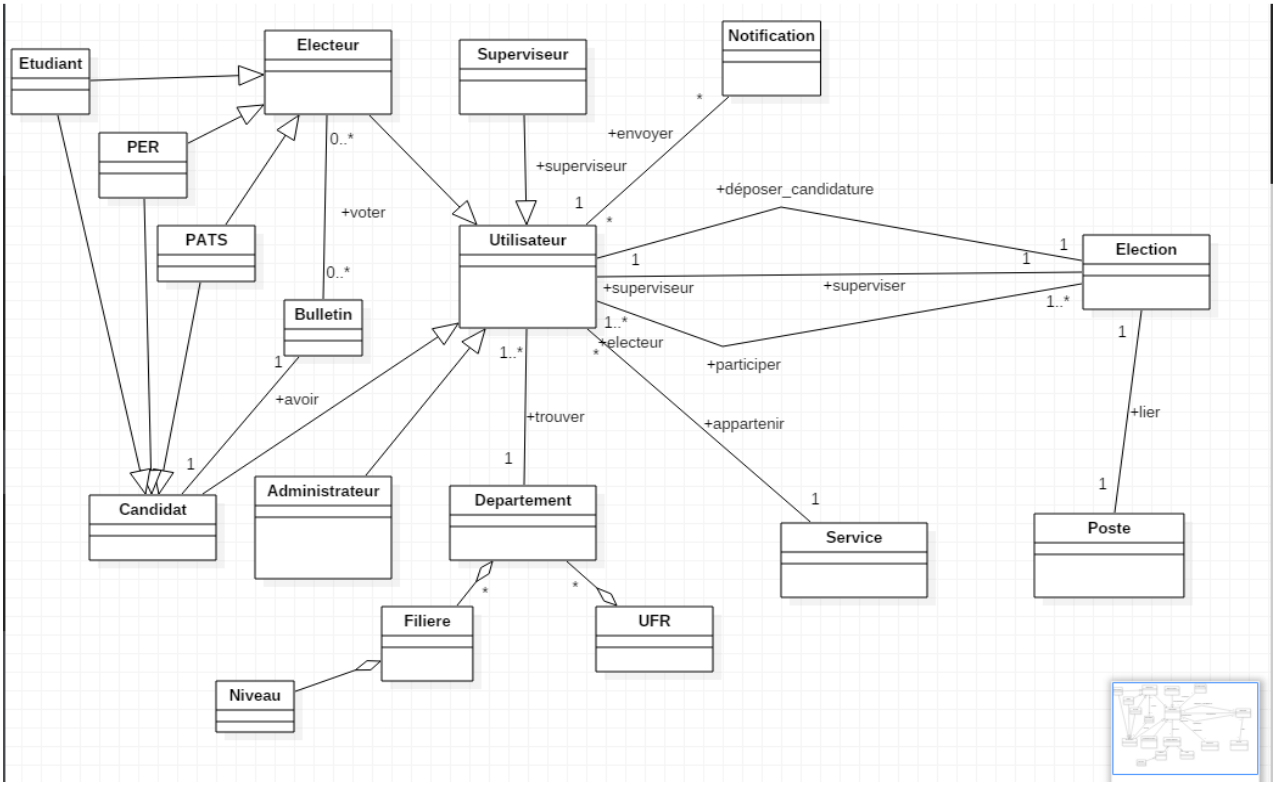


Figure 19: ***Diagramme de classe détaillé de l'application***

CHAPITRE V : IMPLEMENTATION ET PRESENTATION DE L'APPLICATION

Dans ce chapitre, il sera question d'expliquer les grands points de la réalisation de notre système de vote électronique. L'implémentation va s'appuyer essentiellement sur des résultats fournis dans la partie conception. La réalisation du système implique un choix minutieux des technologies, des langages à utiliser et du type de base de données.

Nous essayerons de détailler la partie implémentation en présentant l'ensemble des outils utilisés et leur application sur notre projet et enfin présenter les principales fonctionnalités de notre application.

I. *Implémentation de l'application*

Dans cette partie, nous allons présenter les différents outils utilisés pour l'implémentation de notre système.

1. *Les serveurs utilisés pour notre application*

a. *SGBD (MySQL)*

MySQL est un système de gestion de bases de données relationnelles (SGBDR). Il est distribué sous une double licence GPL et propriétaire. Il fait partie des logiciels de gestion de bases de données les plus utilisés au monde, autant par le grand public (applications web principalement) que par des professionnels, en concurrence avec Oracle, PostgreSQL et Microsoft SQL Server. C'est un serveur de bases de données relationnelles développé dans un souci de performances élevées en lecture, ce qui signifie qu'il est davantage orienté vers le service de données déjà en place que vers celui de mises à jour fréquentes et fortement sécurisées. Il est multi-thread et multi-utilisateur.

C'est un logiciel libre open source, développé sous double licence selon qu'il est distribué avec un produit libre ou avec un produit propriétaire. Dans ce dernier cas, la licence est payante, sinon c'est la licence publique générale GNU (GPL) qui s'applique. Un logiciel qui intègre du code MySQL ou intègre MySQL lors de son installation devra donc être libre ou acquérir une licence payante. Cependant, si la base de données est séparée du logiciel propriétaire qui ne fait qu'utiliser des API tiers (par exemple en C# ou PHP), alors il n'y a pas besoin d'acquérir une licence payante MySQL. Ce type de licence double est utilisé par d'autres produits comme le Framework de développement de logiciels Qt **[5].**

b. *Serveur d'application (Apache Tomcat embarqué)*

Tomcat est un serveur HTTP à part entière. De plus, il gère les servlets et les JSP. Le JSP est un moteur de template permettant de gérer la vue des données rendues par le Servlet. Tomcat a été écrit en langage Java. Il peut donc s'exécuter via la machine virtuelle Java sur n'importe quel système d'exploitation la supportant.

2. *Les technologies utilisées à l'implémentation*

a. *Technologies utilisées pour la partie Back-end*

i. *Spring Framework*

Spring est conçu comme une sorte de boîte à outils, au contraire d'autres Framework. Les modules peuvent être utilisés de façon indépendante. Spring est d'ailleurs disponible sous deux formes, celle d'un jar (librairie Java) unique et celle de plusieurs fichiers jar permettant de ne rajouter au projet que la partie que l'on souhaite utiliser (i.e. Spring Core, Spring Remoting, etc.).

D'autre part, Spring fournit non seulement des services de type fonctionnel comme par exemple les transactions mais est également utile d'un point de vue conceptuel en améliorant la qualité du design. Cela par l'utilisation quasi-systématique d'interface.

ii. *Spring boot*

Spring Boot est un projet ou un micro Framework qui a notamment pour but de faciliter la configuration d'un projet Spring et de réduire le temps alloué au démarrage d'un projet. Pour arriver à remplir cet objectif, Spring Boot se base sur plusieurs éléments :

Un site web (https://start.spring.io/) qui vous permet de générer rapidement la structure de votre projet en y incluant toutes les dépendances Maven nécessaires à votre application. Cette génération est aussi disponible via le plugin Eclipse STS.

L'utilisation de « Starters » pour gérer les dépendances. Spring a regroupé les dépendances Maven de Spring dans des « méga dépendances » afin de faciliter la gestion de celles-ci. Par exemple si vous voulez ajouter toutes les dépendances pour gérer la sécurité il suffit d'ajouter le starter « spring-boot-starter-security ».

L'auto-configuration, qui applique une configuration par défaut au démarrage de votre application pour toutes dépendances présentes dans celle-ci. Cette configuration s'active à partir du moment où vous avez annoté votre application avec « @EnableAutoConfiguration » ou « @SpringBootApplication ». Bien entendu cette configuration peut être surchargée via des

propriétés Spring prédéfinie ou via une configuration Java. L'auto-configuration simplifie la configuration sans pour autant vous restreindre dans les fonctionnalités de Spring. Par exemple, si vous utilisez le starter « spring-boot-starter-security », Spring Boot vous configurera la sécurité dans votre application avec notamment un utilisateur par défaut et un mot de passe généré aléatoirement au démarrage de votre application.

En plus de ces premiers éléments qui facilitent la configuration d'un projet, Spring Boot offre d'autres avantages notamment en termes de déploiement applicatif. Habituellement, le déploiement d'une application Spring nécessite la génération d'un fichier .war qui doit être déployé sur un serveur comme un Apache Tomcat. Spring Boot simplifie ce mécanisme en offrant la possibilité d'intégrer directement un serveur Tomcat dans votre exécutable. Au lancement de celui-ci, un Tomcat embarqué sera démarré afin de faire tourner votre application **[8].**

Figure 20:***Spring boot par rapport à Spring***

iii. Thymeleaf

Thymeleaf est un moteur de Template Java moderne côté serveur pour les environnements web et autonomes. L'objectif principal de Thymeleaf est d'apporter des modèles naturels élégants à votre flux de travail de développement - HTML qui peut être correctement affiché dans les navigateurs et aussi travailler comme des prototypes statiques, permettant une collaboration plus forte dans les équipes de développement.

Avec des modules pour Spring Framework, une multitude d'intégrations avec vos outils préférés, et la possibilité de brancher vos propres fonctionnalités, Thymeleaf est idéal pour le développement web JVM HTML5 moderne **[9].**

b. Technologies utilisées pour la partie Front-End

i. Bootstrap

Bootstrap est une collection d'outils utiles à la création du design (graphisme, animation et interactions avec la page dans le navigateur ... etc. ...) de sites et d'applications web. C'est un ensemble qui contient des codes HTML et CSS, des formulaires, boutons, outils de navigation et autres éléments interactifs, ainsi que des extensions JavaScript en option. C'est l'un des projets les plus populaires sur la plate-forme de gestion de développement GitHub **[10].**

ii. Font-awesome

Font Awesome est un toolkit de polices et d'icônes basé sur CSS et LESS. Il a été conçu par Dave Gandy pour être utilisé avec Bootstrap, puis a été intégré au BootstrapCDN. Font Awesome a une part de marché de 20% parmi les sites Web qui utilisent des scripts de polices tiers sur leur plate-forme, ce qui la place au deuxième rang derrière Google Fonts.

La police Awesome 5 est sortie le 7 décembre 2017 avec 1 278 icônes. La version 5 est proposée en deux versions: Font Awesome Free et Font Awesome Pro (disponible moyennant un supplément). Les versions gratuites (toutes les versions jusqu'à 4 et la version gratuite pour 5) sont disponibles sous licence de licence ouverte Open Font 1.1, Créative Commons Attribution 4.0 et licence MIT **[11].**

iii. JQuery

JQuery est une bibliothèque JavaScript conçue pour simplifier les scripts HTML côté client. Il s'agit d'un logiciel gratuit à code source ouvert utilisant la licence permissive MIT. L'analyse Web indique qu'il s'agit de la bibliothèque JavaScript la plus largement déployée.

La syntaxe de jQuery est conçue pour faciliter la navigation dans un document, sélectionner des éléments DOM, créer des animations, gérer des événements et développer des applications Ajax. JQuery offre également aux développeurs des fonctionnalités leur permettant de créer des plug-ins à partir de la bibliothèque JavaScript. Cela permet aux développeurs de créer des abstractions pour une interaction et une animation de bas niveau, des effets avancés et des widgets de haut niveau et utilisables pour le même thème. L'approche modulaire de la bibliothèque jQuery permet la création de pages Web dynamiques et d'applications Web puissantes **[12].**

3. *Les outils utilisés pour l'implémentation*

a. *Intellij Idea*

Intellij IDEA est un environnement de développement intégré Java (IDE) pour le développement de logiciels. Il est développé par JetBrains (anciennement connu sous le nom d'Intellij) et est disponible sous forme d'édition communautaire Apache 2 sous licence et dans une édition commerciale propriétaire. Les deux peuvent être utilisés pour le développement commercial **[13].**

b. *Star UML*

StarUML est un outil de modélisation adaptée à UML 2 de MKLab. La licence du logiciel était sous une version modifiée de la GNU GPL jusqu'en 2014, date à laquelle une version réécrite 2.0.0 a été publiée pour des tests bêta sous une licence propriétaire.

Après avoir été abandonné pendant un certain temps, le projet a repris le dessus pour passer de Delphi à Java / Eclipse, puis s'être à nouveau arrêté. En 2014, une version réécrite a été publiée sous licence propriétaire **[14].**

c. *L'outil Git*

Git est un logiciel de gestion de versions décentralisé. C'est un logiciel libre créé par Linus Torvalds, auteur du noyau Linux, et distribué selon les termes de la licence publique générale GNU version 2. En 2016, il s'agit du logiciel de gestion de versions le plus populaire qui est utilisé par plus de douze millions de personnes. Il permet de gérer l'historique d'avancement de votre projet et de faire des retours en arrière sur une partie de votre projet enregistré **[15].**

4. *L'arborescence des fichiers de l'application*

Les projets créés sous spring boot ont en général la même arborescence sur les fichiers générés :
Le dossier. idea : Il contient un sous dossier librairies et un ensemble de fichiers propres au configuration du projet sous Intellij. En général, ce dossier n'est pas pris en compte au cours du développement.
Le dossier .mvn : Ce dossier contient les utilitaires Maven nécessaire au bon fonctionnement de notre projet.
Le dossier src : C'est le dossier où nous aurons à travailler. L'ensemble des fichiers qui compose l'application sera défini au niveau de dossiers. Nous pouvons en son sein créer tout un ensemble de sous dossiers nécessaires pour l'implémentation. Le dossier src contient plusieurs répertoires que sont :

- **src/main/java :** tous nos fichiers java implémentés sont dans ce répertoire. Il constitue la partie Back-end de l'application.
- **src/main/resources :** Il contient toutes les ressources dont l'application a besoin. Il gère essentiellement la partie front-end de l'application.
- **src/test/java :** Il gère les tests unitaires effectués .

Le dossier target : Il contient l'ensemble des fichiers exécutables après compilation.

Le dossier external librairies : il contient l'ensemble des fichiers nécessaires au fonctionnement de notre application.

Le fichier POM (appelé Project Object Model) est l'un des plus important sur un projet spring. Il permet de détailler toute la configuration du projet en gérant l'ensemble des dépendances dont a besoin l'application. Il donne des informations sur la version spring utilisé, les ressources, les tests etc…. Il est sous la forme d'un fichier XML bien structuré et se trouve dans le répertoire de base de l'application. La figure ci-dessous permet d'avoir un aperçu de l'arborescence.

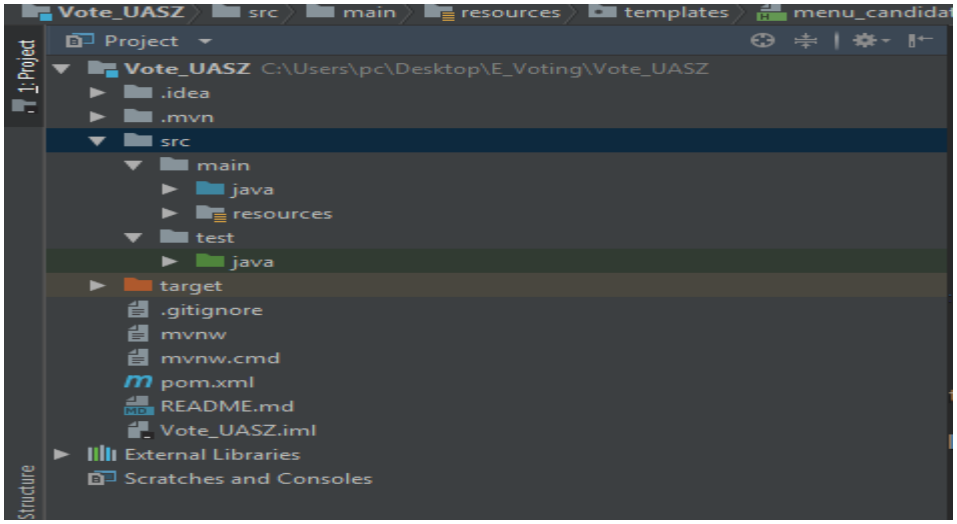


Figure 21: ***Arborescence des fichiers de l'application***

5. *L'arborescence des packages de l'application*

L'organisation du projet en package permet de mieux scinder les sous parties du projet. Elle permet d'avoir une meilleure lisibilité et une meilleure approche de l'architecture 3 tiers tout en respectant le modèle MVC. Les packages permettent d'avoir un code indépendant et assure la réutilisabilité du code sur d'autres projets. Dans notre cas, nous avons un ensemble de packages organisés respectivement pour le back-end et le front-end

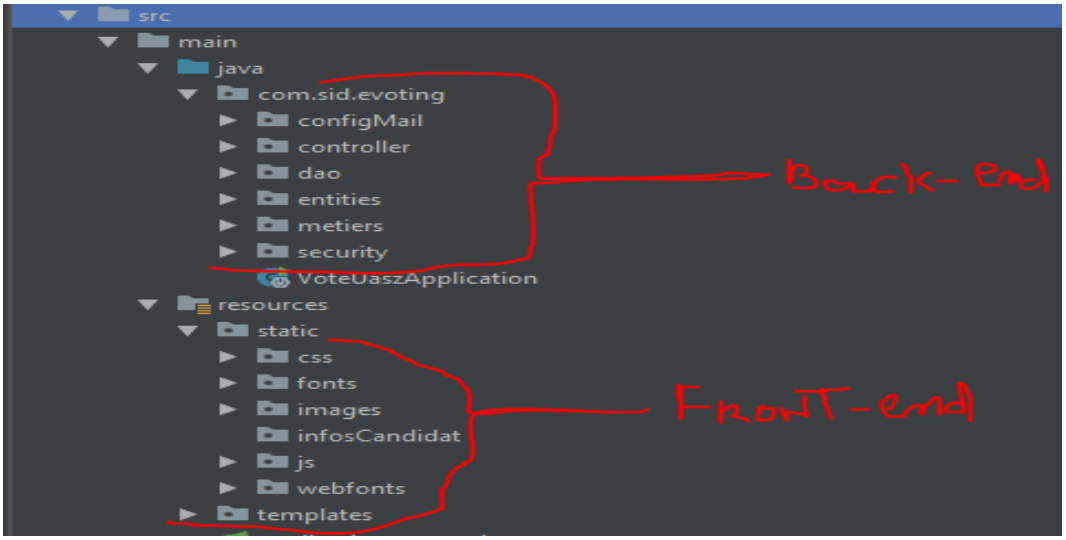


Figure 22: ***L'arborescence des packages de l'application***

a. *Les packages de la partie Back-end*

i. *Entities*

Il constitue l'un des packages les plus important car il permet de définir l'ensemble des entités agissant au niveau de notre application. Les entités crées sont ensuite générées en table au niveau de notre base de données grâce à des dépendances telles que JPA et Hibernate. Ainsi, nous avons un ensemble d'annotations propres à la construction de tables.

@Entity : C'est l'annotation dont on ne peut pas se passer pour créer notre table sur la base de données. Elle indique à JPA de persister la classe définie. Si cette annotation est ignorée, alors toutes les configurations suivantes seront ignorées aussi.

@Table : Elle n'est pas obligatoire. Elle permet d'indiquer le nom principal de la table généré grâce l'argument **name.** Si cette annotation est absente, la table portera le même nom que celui de la classe.

@Id : Il désigne essentiellement la clé primaire de la table. Elle est obligatoire

@Column : Elle est facultative, elle permet d'indiquer le nom de la colonne grâce à **name** et de gérer certaines contraintes comme l'unicité grâce à l'argument (**Unique=true**). La contrainte unique permet d'éviter des risques de doublons sur les valeurs enregistrées. Nous pouvons avoir d'autres annotations telles que **@NotNull** qui indique que la valeur de l'attribut ne doit pas être nulle.En illustration, nous pouvons voir le code ci-dessous de la classe générique « User »

```
@Entity
@Table(name="user")
@Inheritance(strategy = InheritanceType.SINGLE_TABLE)
@DiscriminatorColumn(name="type_user",discriminatorType = DiscriminatorType.STRIN
@JsonIgnoreProperties({"hibernateLazyInitializer", "handler"})
public class User implements Serializable {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long idUser;
    @Column(name = "username",unique = true)
    @NotNull
    private String username;
    private String name;
    private boolean active;
    private String password;
    private String profession;
    private String adresseU;
    private String telephoneU;
```

Figure 23: ***Exemple de la classe "User"***

ii. Data Access Object (DAO)

Le package « dao » permet de définir l'ensemble des interfaces dont nous aurons besoin au niveau de l'application. Il nous permet d'accéder à l'ensemble des méthodes techniques dont nous aurons besoin. Pour accéder à certaines méthodes créées sur Spring Data, il faudra que notre interface hérite de **« JPaRepository »** où nous indiquerons le nom de la classe, et le type de la clé primaire.

Le code ci-dessous, illustre l'interface **« CandidatRepository »**

```
public interface CandidatRepository extends JpaRepository<Candidat, Long> {
  //rechercher les candidats a partir du username
    @Query("select c from Candidat c where c.username like:mc")
    public Page<Candidat> chercherParMc(@Param("mc") String mc, Pageable pageable);
    //les candidats inscrits
    @Query("select c from Candidat c where c.election.idElection=:ide")
    public Page<Candidat> getCandidats(@Param("ide") Long ide, Pageable pageable);

    //avoir une list de Candidats inscrits
    @Query("select c from Candidat c, Election e where c.election.idElection=e.idE
    public List<Candidat> getCandidatsE();

    //les candidats lies à un departement
    @Query("select c from Candidat c, Departement d where c.departement.idDep=d.idD
    public List<Candidat> listCandidatDep();
```

*Figure 24:**Exemple d'interface "CandidatRepository"***

iii. Métiers

Le package **« métiers »** permet de gérer la logique fonctionnelle de l'applications. Toutes les fonctions dont nous avons besoin sont définies sur ce répertoire. Le package « métiers » est organisé en deux sous packages que sont **« interfaces »** et **« impl ».** Le package **« interfaces »** gère toutes les interfaces métiers alors que le package **« impl »** gère toutes les classes implémentant les interfaces.

Le code ci-dessous illustre l'interface **« IBulletin »**

```
public interface IBulletin {
    //to save bulletin candidat
    public Bulletin saveBulletin(Bulletin bulletin);

    //list bulletin
    public List<Bulletin> listBulletins();

    //to get bulletin by id
    public Bulletin getBulletin(Long idBulletin);

    //to count
    public List<Bulletin> compterVoixForCandidat();
    // list
    public List<Candidat> listCandidatsBulltin();
    //list bulletin candidat
    public List<Candidat> listCandidatsBulletinDep(Long id);

    public List<Candidat> listCandidatBuDepF(Long idf, Long idn);
```

Figure 25:exemple interface "IBulletin"

La classe implémentant l'interface doit porter l'annotation **@Service**. Elle sera ainsi prise en compte. L'annotation **@Transactional** n'est pas obligatoire. Elle permet de gérer toutes les transactions au niveau de la base de données.

Le code ci-dessous illustre la classe **« BulletinImpl »**

```
@Service
public class BulletinImpl implements IBulletin {

    @Autowired
    private BulletinRepository bulletinRepo;
    @Override
    public Bulletin saveBulletin(Bulletin bulletin) {
        return bulletinRepo.save(bulletin);
    }

    @Override
    public List<Bulletin> listBulletins() {
        return bulletinRepo.findAll();
    }

    @Override
    public Bulletin getBulletin(Long idBulletin) {
        return bulletinRepo.getOne(idBulletin);
    }
BulletinImpl > nombreVoixCandidat()
```

Figure 26:Exemple de la classe "BulletinImpl"

iv. Controllers

Le package « controller » permet de gérer l'ensemble des contrôleurs de l'application. C'est la partie intermédiaire entre la partie web et les interfaces.

@Controller : il permet de marquer une classe comme étant un controller Spring MVC

@Autowired : c'est une annotation qui permet d'assurer l'injection des dépendances. Pour accéder à l'ensembles des méthodes écrites au niveau de l'interface, il faut appeler l'interface concernée en l'instanciant comme un élément Bean de Spring

@GetMapping ou @RequestMapping : Il permet de spécifier l'appel d'une méthode grâce à une Uri donnée.

Le code ci-dessous permet d'illustrer le controller **« BulletinController »**

```
@Controller
public class BulletinController {
    @Autowired
    private IUser iUser;
    @Autowired
    private IBulletin bulletinM;
    @Autowired
    private IElecteur iElecteur;
    //le nombre de voix
    @GetMapping("/compterForCandidat")
    @ResponseBody
    public List<Bulletin> compteForCandidat(Model model){
        return bulletinM.compterVoixForCandidat();
    }

    @GetMapping("/candidatBulletin")
    @ResponseBody
    public List<Candidat> listCandidatB(Model model){
        return bulletinM.listCandidatsBulltin();
```

*Figure 27:**exemple de controller "BulletinController"***

v. *Security*

Dans ce package, nous allons gérer l'ensemble de la sécurité de l'application. L'authentification, les contrôles d'accès, les redirections à partir des rôles sont gérées à ce niveau.

@EnableWebSecurity est une annotation qui permet d'activer toutes les fonctionnalités nécessaires à la configuration. L'annotation **@Configuration** permet d'indiquer à Spring que la classe doit être instanciée comme un Bean au démarrage.

@EnableGlobalMethodSecurity est une annotation qui permet de gérer l'accès à des méthodes grâce au rôle de l'utilisateur avec l'argument **« SecuredEnable = True ».**

L'extension à la classe « **WebSecurityConfigurerAdapter »** fournit tous les services nécessaires. Dans les services assurés, nous avons :

- **L'authentification** : qui gère l'identification de l'utilisateur et de son accès aux différentes ressources.
- **Le contrôle d'accès** : il permet l'accès aux ressources aux personnes autorisées

- **L'intégrité** : A ce niveau, on va s'assurer que les données envoyées ont été bien reçues et qu'elles n'ont pas été modifiées. (Évitant ainsi l'attaque de type Man in the Middle)
- **La confidentialité** : On pourra éviter la lecture des données à des personnes non autorisées

Le code ci-dessous illustre la classe **« SecurityConfig »**

```
@Configuration
@EnableWebSecurity
@EnableGlobalMethodSecurity(securedEnabled = true)
public class SecurityConfig extends WebSecurityConfigurerAdapter {
    @Autowired
    private SimpleAuthenticationSuccess success;
    @Autowired
    private BCryptPasswordEncoder encoder;
    @Autowired
    public void globalConfig(AuthenticationManagerBuilder builder, DataSource dataSource)th
        //builder.inMemoryAuthentication().withUser("admin").password("{noop}1234").roles("
        //builder.inMemoryAuthentication().withUser("prof1").password("{noop}1234").roles("

        builder.jdbcAuthentication()
                .dataSource(dataSource)
                .usersByUsernameQuery("select username as principal, password as credentials
                .authoritiesByUsernameQuery("select user.username as principal, role.role_na
                .rolePrefix("ROLE_")
                .passwordEncoder(encoder);
```

Figure 28: ***Exemple de la classe "SecurityConfig"***

Pour mieux renforcer la sécurité des échanges de données, nous avons aussi mis en place le protocole HTTPS (HyperText Transport Protocol Secure) qui permet le cryptage des données échangées entre le serveur et le navigateur client grâce à l'utilisation des algorithmes symétriques ou symétriques (AES, RSA, BlowFish, SHA256, etc.). Le protocole HTTPS nous permet d'éviter l'attaque de l'homme du milieu (Man In the Middle). Cette attaque consiste donc à intercepter toutes les informations échangées entre le serveur et le client. L'attaquant dispose donc d'outils lui permettant de tromper le serveur et l'application client afin que toutes les informations envoyées par ces deux soient transmises direct dans son terminal. L'attaquant pourra donc exploiter des données sensibles non cryptées. Pour pouvoir l'utiliser de façon optimale, il faudra activer la navigation en mode TLS (Transport Layer Secure) depuis l'invite de commande. Le protocole HTTPS peut être activé en local, mais il faut acheter le certificat d'utilisation pour que notre application puisse être déployée avec le protocole HTTPS.

b. Les packages de la partie Front-end

i. Static

Ce package permet de mettre l'ensemble des éléments qui vont interagir avec les pages web de l'application :

- Le **dossier CSS** renferme tous les fichiers CSS nécessaires (Bootstrap, font-awesome, style, etc…)
- Le **dossier JS** renferme tous les fichiers JS nécessaires (JQuery, main, etc…)
- Le **dossier Images** contient toutes les images
- Le **dossier Fonts** et **WebFonts** contient toutes les polices et icônes nécessaires aux fichiers CSS

ii. Templates

Ce package contient l'ensemble de nos pages web crées c'est-à-dire tous les fichiers HTML et qui seront utilisées dans le projet.

II. Présentation de l'application

À travers les différentes phases d'étude menées, nous avons pu développer une application de vote électronique. Ce système constitue un prototype et couvre l'ensemble du processus d'une élection. Le système présente un ensemble de fonctionnalités et l'accès est permis à quatre types d'utilisateur (l'administrateur, le (s) superviseur (s), les électeurs et le (s) candidat (s)). Les principales fonctionnalités du système sont présentées dans les sections suivantes. Mais avant cela, nous présentons une vue globale de l'application sur la figure suivant.

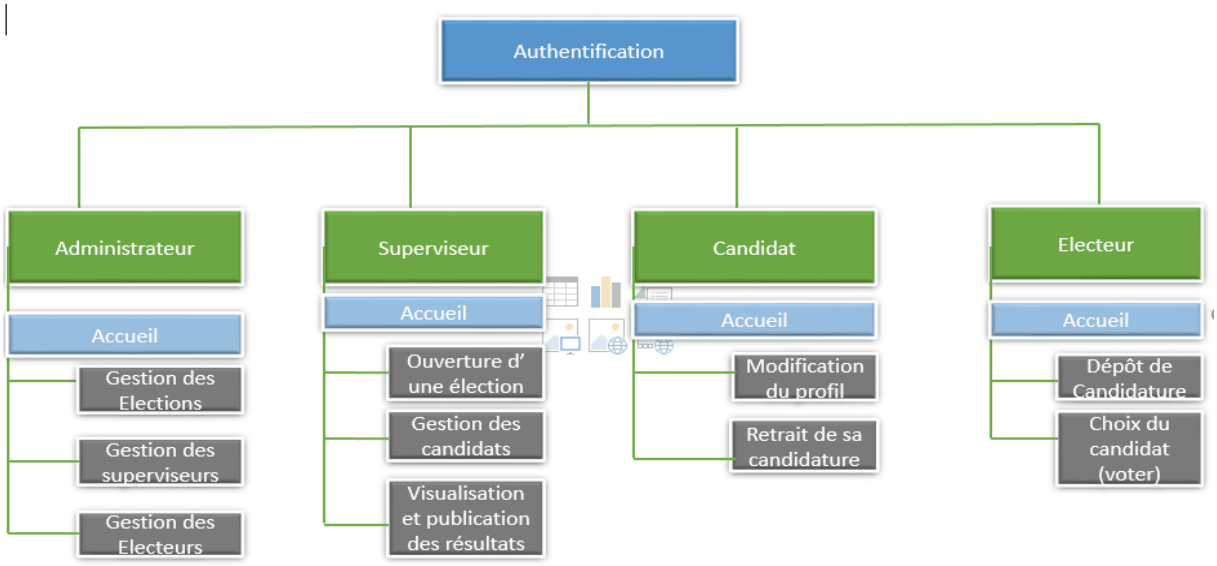


Figure 29: ***Vue globale de l'application***

1. Authentification

L'authentification est la première fonctionnalité implémentée et c'est la première page qui apparaît lorsque vous tentez d'accéder à la plateforme. Elle permet de rediriger à l'utilisateur le menu adéquat en fonction de son profil. L'utilisateur devra saisir son login et son code d'accès. Pour chaque électeur, un code d'accès lui est envoyé dès l'ouverture d'une élection.

Figure 30: ***Page d'authentification du système***

Une fois l'authentification réussie, le système redirige à l'utilisateur le menu adéquat (cela en prenant en compte son rôle).

Nous allons donc présenter le menu administrateur, ensuite expliquer l'ouverture et le dépôt de candidatures, simuler la phase de scrutin et enfin montrer la publication des résultats

2. Menu Administration

C'est dans ce menu que toute l'administration de l'application est gérée. La gestion des électeurs, celle des élections et des superviseurs sont faites à ce niveau.

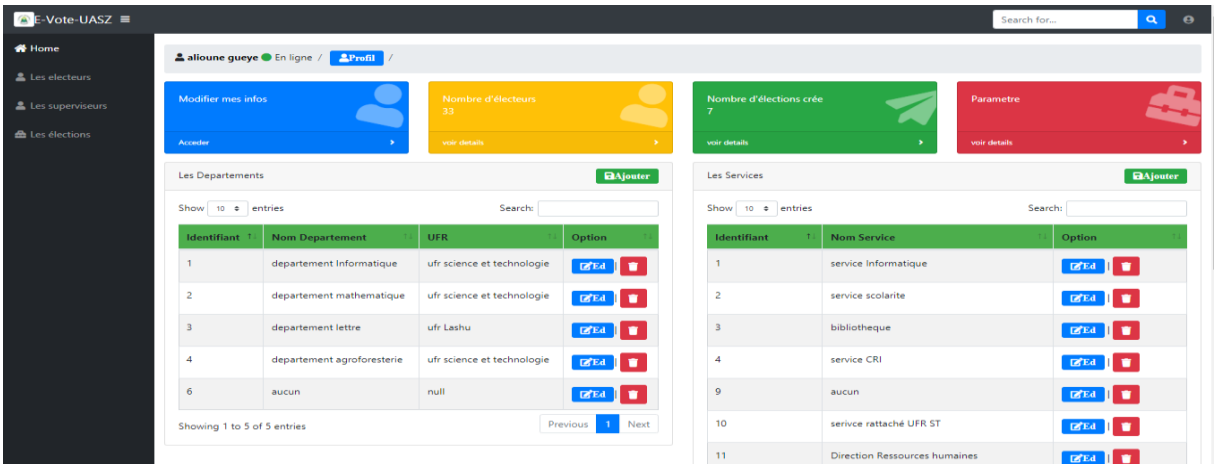


Figure 31 **: Menu administrateur**

3. L'ouverture des élections

Dans cette partie, le superviseur va se charger d'ouvrir l'élection en définissant la date de dépôt de candidatures, ainsi que la date de scrutin. Un mail d'alerte sera envoyé aux électeurs après ouverture.

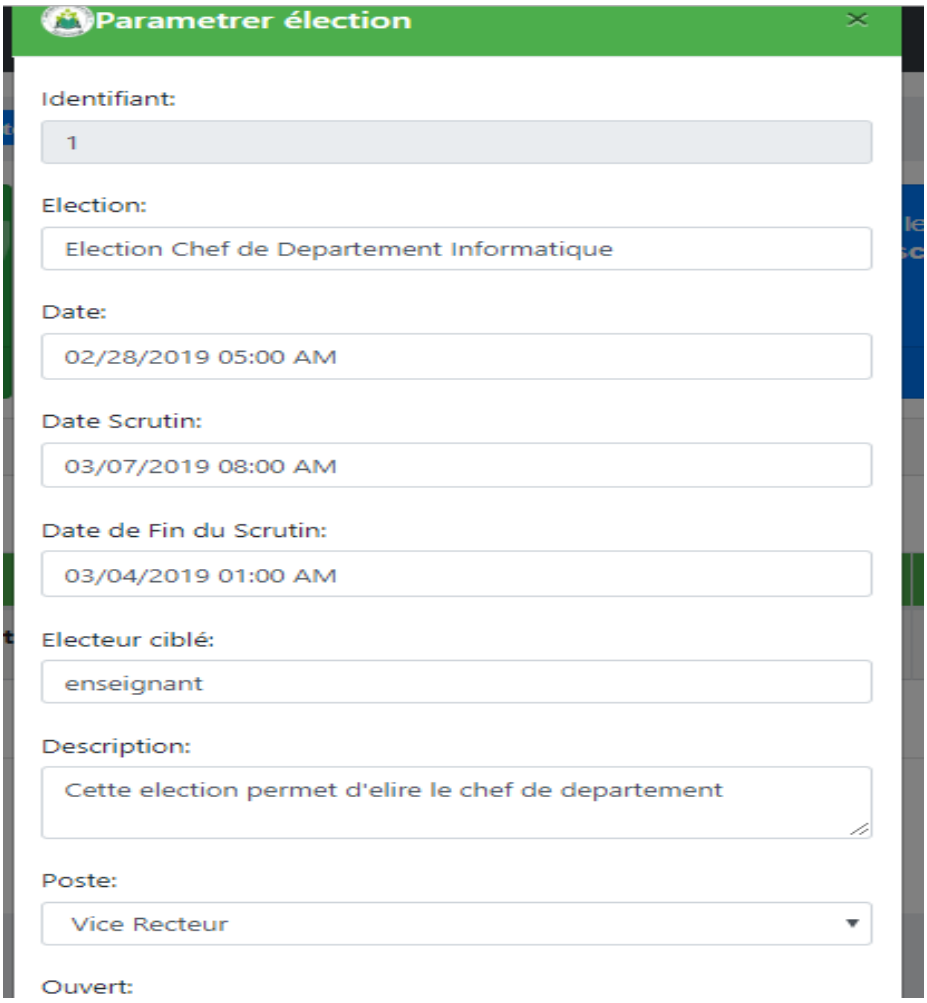


Figure 32: **Ouverture d'une élection**

Le superviseur peut gérer plusieurs élections et paramétrer donc leur ouverture. Les électeurs alertés par la note d'information envoyée par mail, ils peuvent accéder au système grâce au code d'accès envoyé pour éventuellement déposer leur candidature.

4. La phase de dépôt de candidature

Dans cette étape, les électeurs désirant être candidat, peuvent s'inscrire sur le formulaire. Une vérification interne est faite pour vérifier si la personne correspond bien aux critères de candidature.

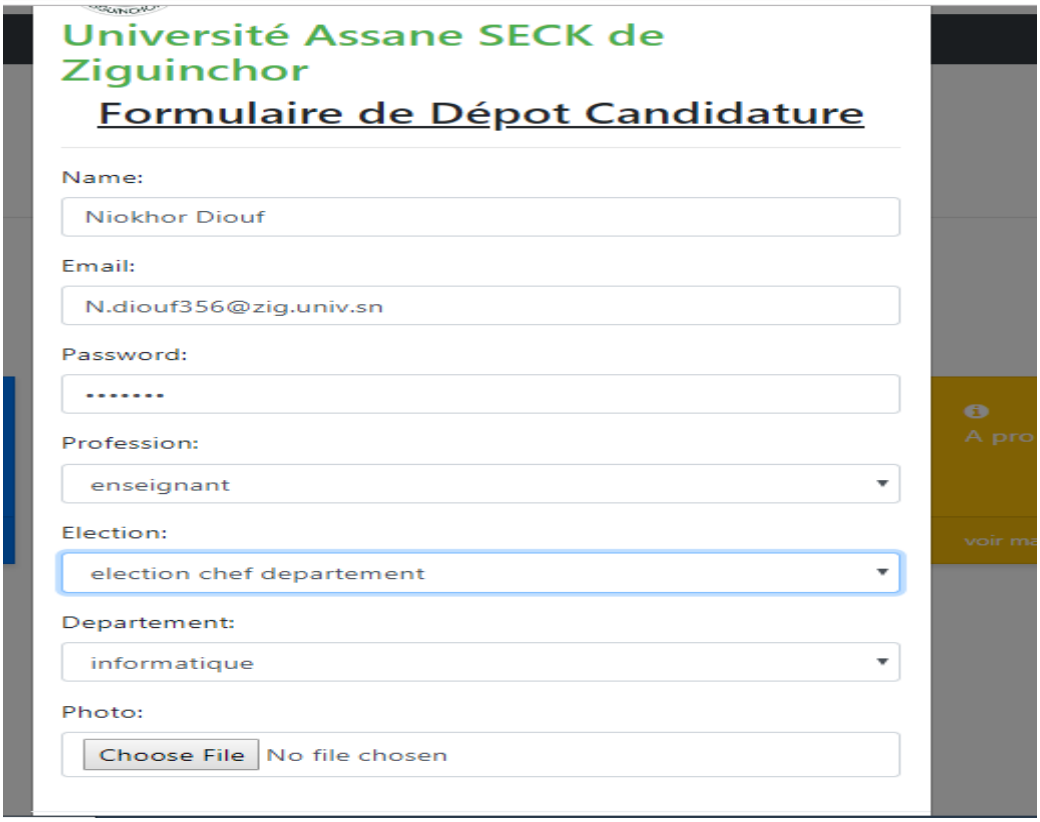


Figure 33: ***Formulaire de dépôt de candidature***

Si la candidature est validée, l'électeur est redirigé vers une page de confirmation où il pourra récupérer ses identifiants de connexion en tant que candidat à l'élection ouverte.

5. *La phase de scrutin*

La phase de scrutin est faite après la clôture des dépôts de candidatures. Les électeurs accèdent à nouveau à la plateforme pour voter. S'il clique sur l'option « effectuer vote », une boîte de dialogue s'affiche et il choisit sur l'une des élections ouvertes dont il participe. La figure ci-dessous montre le procédé.

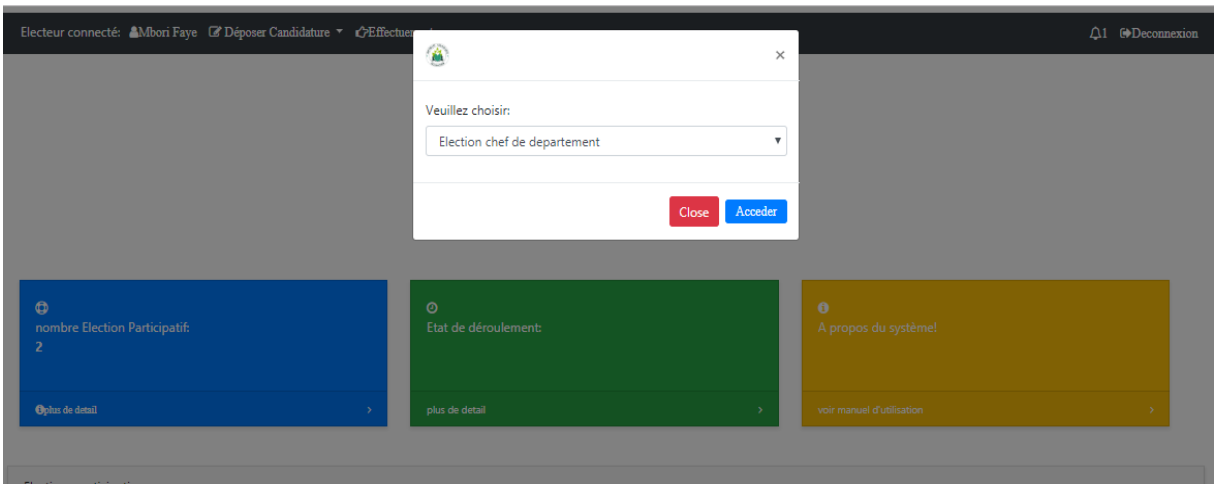


Figure 34: ***Boîte d'option élection***

Après avoir choisi, l'électeur sera redirigé vers la liste des candidats inscrits sur l'élection

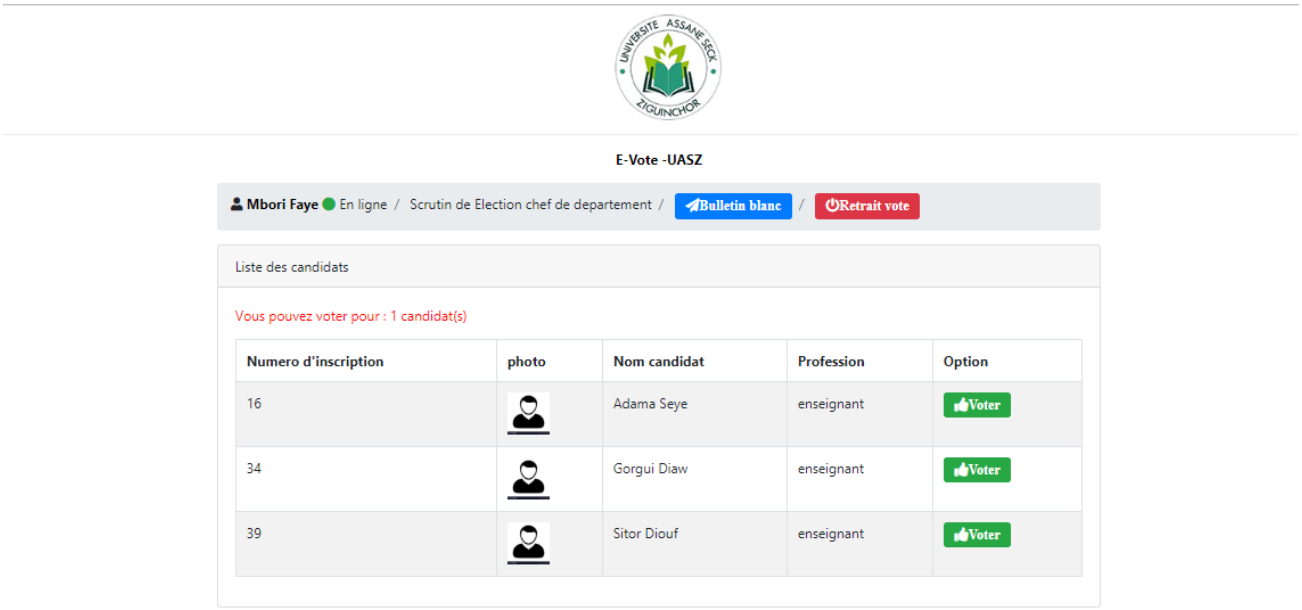


Figure 35: ***Liste des bulletins de candidats***

Pour cette élection, l'électeur peut voter que pour un seul candidat. Des indications sont données sur chaque type d'élection, car il peut arriver dans certaines élections de voter pour plusieurs candidats (le cas de l'élection des membres au conseil d'administration). L'électeur peut aussi décider de s'abstenir dans une élection donnée. Dans cet exemple, si l'électeur choisit le candidat, il sera redirigé par la suite dans une page de confirmation. Il pourra ainsi voir les élections restantes dont il peut participer.

Figure 36: ***Message de confirmation***

Il peut encore retourner au niveau de la plateforme et faire la même démarche pour l'élection restante dont il peut participer

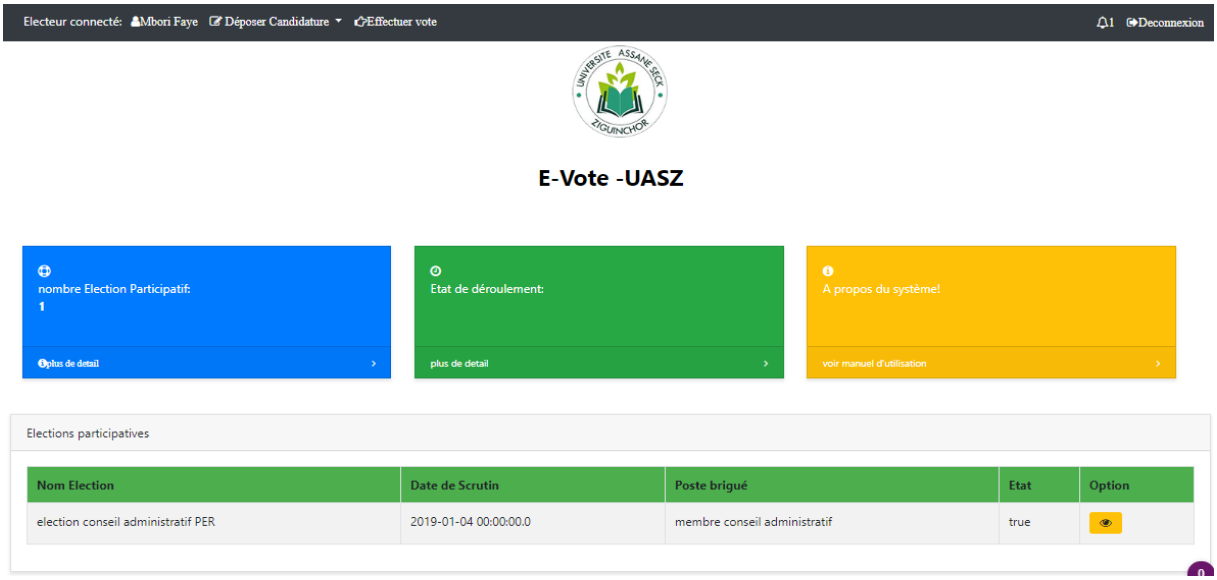


Figure 37: ***Menu électeur avec l'élection restante***

Une fois qu'il termine et qu'il n'y a plus d'élections ouvertes, l'électeur sera redirigé vers la page de sortie de la plateforme où il ne pourra plus avoir accès jusqu'à la réouverture de nouvelles élections.

6. *La phase de supervision et de publication des résultats*

Le déroulement du scrutin est donc suivi en permanence par le superviseur qui surveille le temps de déroulement du vote. Il peut consulter les premières statistiques sur le taux de participation. Les résultats finaux sont visibles à la fin du scrutin.

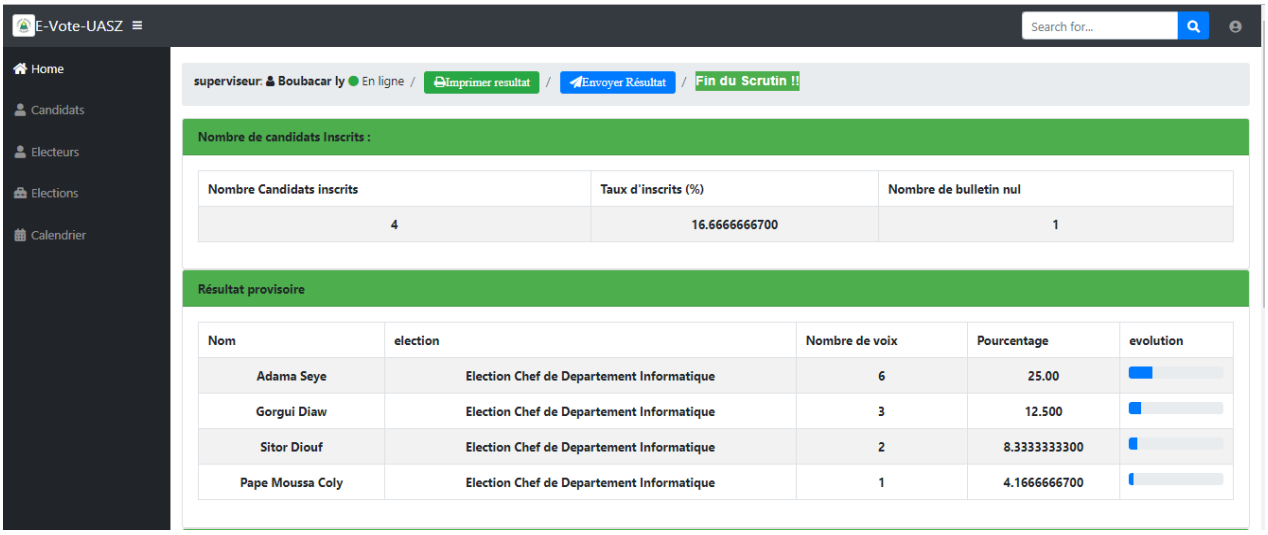
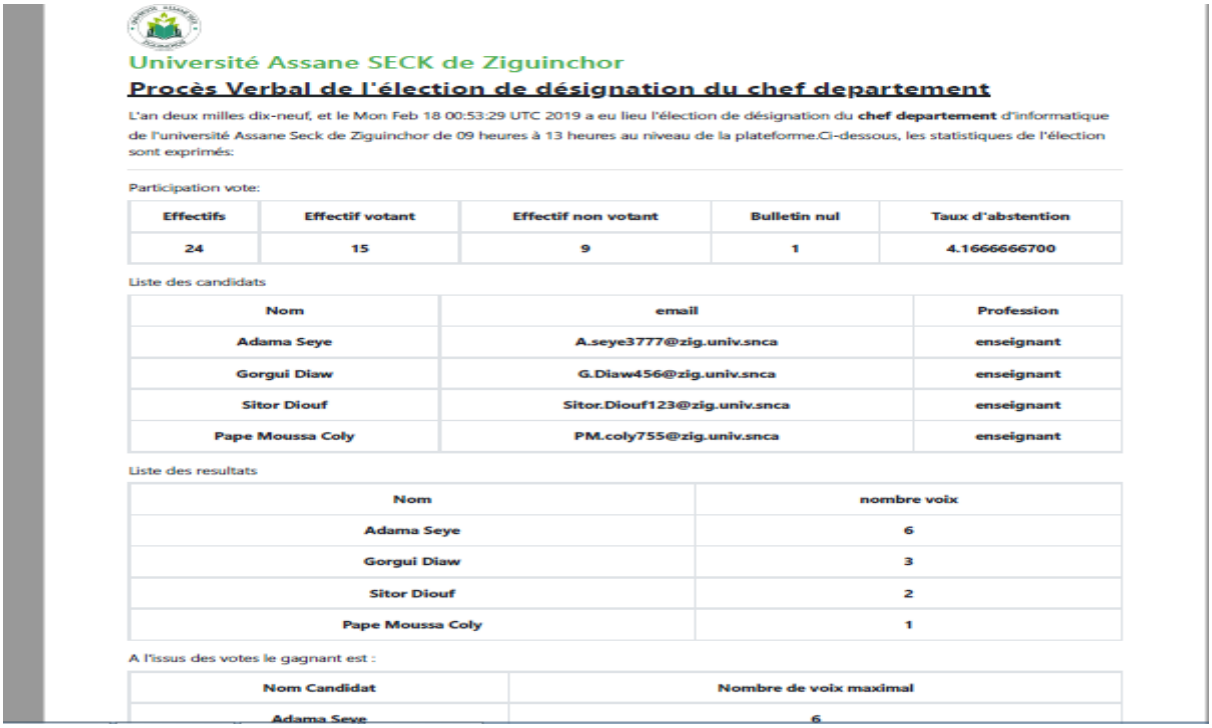


Figure 38: ***Les résultats du scrutin***

Le superviseur pourra imprimer le procès-verbal du scrutin et l'envoyer aux électeurs grâce au système de mail intégré (sur la figure 40).



Université Assane SECK de Ziguinchor

Procès Verbal de l'élection de désignation du chef departement

L'an deux milles dix-neuf, et le Mon Feb 18 00:53:29 UTC 2019 a eu lieu l'élection de désignation du **chef departement** d'informatique de l'université Assane Seck de Ziguinchor de 09 heures à 13 heures au niveau de la plateforme.Ci-dessous, les statistiques de l'élection sont exprimés:

Participation vote:

Effectifs	Effectif votant	Effectif non votant	Bulletin nul	Taux d'abstention
24	15	9	1	4.1666666700

Liste des candidats

Nom	email	Profession
Adama Seye	A.seye3777@zig.univ.snca	enseignant
Gorgui Diaw	G.Diaw456@zig.univ.snca	enseignant
Sitor Diouf	Sitor.Diouf123@zig.univ.snca	enseignant
Pape Moussa Coly	PM.coly755@zig.univ.snca	enseignant

Liste des resultats

Nom	nombre voix
Adama Seye	6
Gorgui Diaw	3
Sitor Diouf	2
Pape Moussa Coly	1

A l'issus des votes le gagnant est :

Nom Candidat	Nombre de voix maximal
Adama Seye	6

Figure 39: ***Procès-verbal du scrutin***

CONCLUSION GENERALE ET PERSPECTIVES

L'objectif de notre était d'étudier et mettre en place un système web de vote électronique adapté à l'ensemble des élections organisées au sein de l'université Assane SECK de Ziguinchor.

Pour ce faire, nous avons essayé tant bien que mal de mettre en place un prototype répondant aux exigences citées par le DRH et le chargé pédagogique de l'UFR Science et Technique. Il permet la gestion ainsi que le suivi du processus de vote pour une élection donnée. Le système essaie donc de gérer l'ouverture et l'envoie de notification aux utilisateurs enregistrés au niveau de la base de données de l'université, de permettre le dépôt de candidatures et d'effectuer le vote et enfin de permettre un suivi des statistiques du scrutin et aussi un envoi des résultats d'élection aux utilisateurs concernés. Une élection ouverte sera déroulée au niveau de la plateforme jusqu'à sa fermeture.

Pour pouvoir mener à bien le travail avec les résultats attendus, nous avons scindé notre mémoire en cinq chapitres. Dans le premier chapitre, nous avons essayé de centrer le sujet en montrant d'abord l'état de l'art des systèmes de vote en ligne sur leur évolution, ensuite la présentation succincte de l'université et des différents types d'élection organisés et enfin des problèmes visibles sur l'organisation des élections. Dans le deuxième chapitre, nous avons essayé de présenter la méthodologie de notre choix qui est SCRUM. Et nous avons essayé de l'appliquer au niveau du troisième chapitre sur l'analyse et les spécifications des besoins fonctionnels définis. De cette analyse faite, nous avons pu recenser l'ensemble des informations nécessaires nous permettant d'élaborer dans le quatrième chapitre notre analyse et conception avec le langage de modélisation UML. Et dans le cinquième chapitre grâce à la conception établie, nous avons pu présenter l'implémentation notre prototype de vote en ligne dédié aux élections de l'université.

En perspective, l'application peut être améliorée pour permettre de mieux couvrir l'ensemble des élections organisées. Elle sera enrichie avec de nouvelles fonctionnalités telles que :

- Renforcer la sécurité et la traçabilité de vote avec l'intégration de la technologie Block Chain en réduisant ainsi les risques d'attaques et de falsifications ;
- L'étendre en mettant en place une solution adaptée aux smartphones ;
- Intégrer entièrement le collège électoral défini pour chaque élection ;

- L'adapter à n'importe quel type d'élection même celles à fort enjeu (élection législative, présidentielle, etc.).

WEBOGRAPHIE

[1] Site de Loria **https://members.loria.fr/VCortier/files/Papers/Bulletin1024-2016.pdf** consulté 11/05/2018

[2] Site de agiliste https://agiliste.fr/guide-de-demarrage-scrum/ consulté le 08/06/2018

[3] Site de Openclassroom https://openclassrooms.com/fr/courses/4670706-adoptez-une-architecture mvc-en-php/4678736-comment-fonctionne-une-architecture-mvc consulté le 15/08/2018

[4] Site de Supinfo https://www.supinfo.com/articles/single/6437-fonctionnement-une-architecture-trois-tiers consulté 15/08/2018

[5] Site de Spring https://translate.google.sn/translate?hl=fr&sl=en&u=http://projects.spring.io/spring-data/&prev=search consulté 16/08/2018

[6] Site de Wikipedia https://fr.wikipedia.org/wiki/MySQL consulté 16/08/2018

[7] Site de Wikipedia https://fr.wikipedia.org/wiki/Apache_Tomcat consulté 16/08/2018

[8] Site du forum https://zekey.developpez.com/articles/spring/ consulté le 18/08/2018

[9] Site de groupeafg http://www.groupeafg.com/spring-boot-kesako/ consulté le 18/08/2018

[10] Site de https://fr.wikipedia.org/wiki/Bootstrap_ (framework) consulté le 18/08/2018

[11] Site de Wikipedia https://fr.wikipedia.org/wiki/font_Awesome consulté le 18/08/2018

[12] Site de https://translate.google.sn/translate?hl=fr.wikipedia.org%2Fwiki%2FJQuery consulté le 18/08/2018

[13] Site de https://translate.google.sn/translate?hl=fr.wikipedia.org/FIntelliJ_IDEA consulté le 18/08/2018

[14] Site de https://translate.google.sn/translate?hl=fr.wikipedia.org/wiki/StarUML consulté le 18/08/2018

[15] Site edutechwiki https://edutechwiki.unige.ch/fr/Git_et_Github consulté 18/08/2018