Cahier des charges

BLONDEAU Roman – MENETTRIER Nicolas

**Projet Personnel “GAZETTE”**

Epitech Promo 2020

Sommaire

[Description générale du projet 2](#_Toc525915263)

[Technologies utilisées 2](#_Toc525915264)

[Scope / Domaine du projet 2](#_Toc525915265)

[Description des différentes parties du projet (modules intégrés) 3](#_Toc525915266)

[Découpage des tâches fonctionnelles 3](#_Toc525915267)

[Annexes, documentations et références 6](#_Toc525915268)

# Description générale du projet

*Gazette* est une application desktop open-source permettant de réaliser des newsletters et plus globalement des emails.

# Technologies utilisées

Développement de l’application : (liens en annexes)

* ElectronJS
  + React
  + NodeJS
  + HTML / CSS / JavaScript
  + Electron-store

Développement continu :

* Git
* GitHub

Organisationnel :

* AirTable

# Scope / Domaine du projet

Les différents domaines auxquels le projet se réfère sont les suivants :

* Emailing
* Délivrabilité
* Analyse des données (tracking / analytics)
* Technologies web
* Spam
* Rédaction
* Conception

Plus généralement le projet se rapproche du domaine de la création et d’envois d’email d’information ou publicitaire.

Vous trouverez en annexe des sources à propos de la délivrabilité des emails.

# Description des différentes parties du projet (modules intégrés)

Les différents modules du projet se découpent de la façon suivante :

* Partie 1 : Installation de l’application
* Partie 2 : Système de session utilisateur / profil
* Partie 3 : Interface de planification d’envoie des emails
* Partie 4 : Interface de création des emails
* Partie 5 : Stockage persistant
* Partie 6 : UI/UX (interface utilisateur et expérience utilisateur)

# Découpage des tâches fonctionnelles

Pour le détail de la charge de travail par personne des tâches fonctionnelles, merci de vous référer au fichier Excel « Répartition des tâches » fournis avec ce cahier des charges.

|  |  |
| --- | --- |
| Partie 1 : Installation de l’application |  |
| Tâche fonctionnelle | **Description** |
| Recherche sur l'installation multiplateformes | Le projet doit pouvoir être déployé sur les plateformes Windows, MacOs X ou Linux. |
| Mise en place de l'installation multiplateformes | L’utilisateur doit disposer d’un installateur dédié selon son OS.   * **.dmg** pour MacOs X * **.exe** pour Windows * **.deb** pour Linux |
| Mise en place d'un environnement de *Dev* et de *Production* | L’environnement de développement permet de voir les modifications effectuées dans le code directement mais il n’est pas optimisé et sécurisé au contraire de l’environnement de production qui se déploie à l’aide d’une installation et ne permet pas la modification des fichiers sources. |
| Mise en place d'un serveur distant servant d'API | Serveur back-end |

|  |  |
| --- | --- |
| Partie 2 : Système de session utilisateur / profil |  |
| Tâche fonctionnelle | **Description** |
| Création/suppression d'un compte utilisateur via un serveur distant | L’utilisateur doit pouvoir créer un compte, la vérification et l’enregistrement des données se fait sur le serveur distant |
| Création de l'interface de sélection de compte utilisateur | L’utilisateur doit pouvoir se connecter à son/ses compte(s) |

|  |  |
| --- | --- |
| Partie 3 : Interface de planification d’envoie des emails |  |
| Tâche fonctionnelle | **Description** |
| Création de l'interface permettant voir les newsletters/emails existants | Interface d’accueil mettant en avant toutes les créations de l’utilisateur. Il peut d’ici, accéder aux différents emails, en créer ou en supprimer. |
| Implémentation d'un calendrier permettant de planifier l'envoie d'un email en différé | Pour chaque email crées, l’utilisateur doit pouvoir disposer d’une interface lui permettant de configurer l’envoie de ses mails (destinataires, objet…) ainsi que la date à laquelle il veut l’envoyer. Il est donc possible de programmer un envoie de mails dans le futur. |
| Implémentation du système informatique permettant l'envoi différer des emails via un serveur distant | Le serveur distant doit s’occuper de bien envoyer à la date sélectionnée les emails concernés (CRON) |

|  |  |
| --- | --- |
| Partie 4 : Interface de création des emails |  |
| Tâche fonctionnelle | **Description** |
| Interface : implémentation du visualiseur HTML permettant de voir graphiquement, visuellement à quoi ressemble le mail | L’utilisateur doit pouvoir visualiser dans son ensemble l’email qu’il est en train de créer. |
| Interface : implémentation de l'éditeur de code HTML permettant d'éditer directement le code source de l'email | L’utilisateur doit pouvoir visualiser et modifier directement le code de l’email s’il a un besoin spécifique |
| Interface : implémentation du sélecteur d'élément (texte, image, bouton…) permettant de placer les éléments sur le visualiseur via un glisser / déposer | L’utilisateur doit pouvoir facilement créer son email à l’aide de différents outils permettant d’insérer des éléments directement dans le visualiseur via du glisser/déposer |
| Interface : Manipulation des attributs graphiques (CSS) de l'élément sélectionné | Chaque élément brut inséré doit pouvoir être modifié d’un point de vue graphique et de contenu |

|  |  |
| --- | --- |
| Partie 5 : Stockage persistant |  |
| Tâche fonctionnelle | **Description** |
| Architecture et déploiement d'une base de données sur le serveur distant | Toutes les données utilisateur seront stockées dans une base de données centralisée sur le serveur distant |
| Stockage en cache (local) de certaines données utilisateur | Certaines données sont trop lourdes ou ne nécessitent pas de requêtes au serveur distant, elles seront donc stockées en local sur la machine de l’utilisateur |
| Stockage sur le serveur distant des données utilisateur | Données sensibles ou non-stockable en local. Permet de retrouver tous ses projets sur n’importe quelle machine |
| Stockage en tant que modèle d'un email existant afin de le réutiliser comme base pour un autre email | L’utilisateur doit pouvoir facilement réutiliser une création en tant que base pour un nouvel email |
| Configuration des routes du serveur distant afin d'envoyer/réceptionner les données utiles de façon sécurisée | Chaque requête doit disposer d’une route spécifique effectuant une action précise. |

|  |  |
| --- | --- |
| Partie 6 : UI/UX (interface utilisateur et expérience utilisateur) |  |
| Tâche fonctionnelle | **Description** |
| Recherche sur la meilleure ergonomie pour la création et l'édition des emails | L’utilisateur doit pouvoir naviguer et créer facilement |

# Annexes, documentations et références

Administratif :

Repository GitHub du projet contenant le Powerpoint de présentation : <https://github.com/RomanBlondeau/gazette>

Technologies :

ElectronJS : <https://electronjs.org/>

React: <https://reactjs.org/>

NodeJS: <https://nodejs.org/en/>

Electron-store: <https://github.com/sindresorhus/electron-store>

AirTable : <https://airtable.com/>

Documentation :

Délivrabilité des emails :

<https://fr.wikipedia.org/wiki/D%C3%A9livrabilit%C3%A9>

<https://docs.campaign.adobe.com/doc/AC/fr/DLV_Gestion_de_la_delivrabilite_A_propos_de_la_delivrabilite.html>