Documentation du projet Hedwige

# Table des matières

[I. Table des matières 2](#_Toc211585832)

[II. Présentation générale 3](#_Toc211585833)

[III. Architecture du projet 3](#_Toc211585834)

[IV. Backend (Node.js) 4](#_Toc211585835)

[V. Frontend (React) 5](#_Toc211585836)

[VI. Infrastructure Docker 6](#_Toc211585837)

[VII. Authentification Google OAuth2 7](#_Toc211585838)

[VIII. API Gmail et Calendar 7](#_Toc211585839)

[IX. Persistance de session 8](#_Toc211585840)

[X. Tests unitaires 8](#_Toc211585841)

[XI. Fichiers .env et sécurité 9](#_Toc211585842)

[XII. Démarrage du projet 10](#_Toc211585843)

[XIII. Pistes d’évolution 10](#_Toc211585844)

[XIV. Démarrage du projet 18](#_Toc211585845)

[XV. Pistes d’évolution 18](#_Toc211585846)

# Présentation générale

**Hedwige** est une application web inspirée de l’univers de *Harry Potter*, permettant à un utilisateur de se connecter via son compte Google (OAuth2) et d’interagir avec :

* 📧 **Gmail** : lecture des derniers e-mails avec aperçu du sujet, de la date et du contenu.
* 📅 **Google Calendar** : affichage des événements à venir à partir d’aujourd’hui.

Le tout est présenté dans une interface magique et immersive, avec gestion de session persistante et possibilité de déconnexion sécurisée.

# Architecture du projet

hedwige/

│

├── backend/

│ ├── index.js # Serveur Express principal

│ ├── package.json

│ ├── tests/

│ │ └── app.test.js # Tests unitaires Jest

│ └── .env # Variables sensibles

│

├── frontend/

│ ├── public/

│ │ ├── index.html

│ │ └── animation\_2D.html # Animation d’attente personnalisée

│ ├── src/

│ │ ├── App.js # Composant principal React

│ │ ├── App.css # Thème "Harry Potter"

│ │ └── api.js # Fonctions d’appel à l’API backend

│ ├── package.json

│ └── .env # Variables d’environnement du front

│

├── docker-compose.yml # Orchestration des conteneurs

└── README.md # Ce document

# Backend (Node.js)

**🧩 Technologies**

* **Node.js / Express**
* **Passport.js** (stratégie Google OAuth2)
* **express-session**
* **connect-mongo**
* **googleapis** (accès Gmail / Calendar)
* **dotenv** pour la configuration
* **Jest + Supertest** pour les tests

**📁 Structure simplifiée**

// index.js

require("dotenv").config();

const express = require("express");

const session = require("express-session");

const MongoStore = require("connect-mongo");

const passport = require("passport");

const { google } = require("googleapis");

const cors = require("cors");

const app = express();

app.use(cors({ origin: process.env.FRONTEND\_URL, credentials: true }));

// --- Sessions persistantes

app.use(session({

secret: process.env.SESSION\_SECRET,

resave: false,

saveUninitialized: false,

store: MongoStore.create({ mongoUrl: process.env.MONGO\_URI })

}));

app.use(passport.initialize());

app.use(passport.session());

require("./passport-config")(passport);

// --- Routes principales

app.get("/auth/google", passport.authenticate("google", {

scope: ["profile", "email", "https://www.googleapis.com/auth/gmail.readonly", "https://www.googleapis.com/auth/calendar.readonly"]

}));

app.get("/auth/google/callback", passport.authenticate("google", {

failureRedirect: process.env.FRONTEND\_URL

}), (req, res) => res.redirect(`${process.env.FRONTEND\_URL}/dashboard`));

app.get("/logout", (req, res) => {

if (req.logout) {

req.logout(() => {

req.session.destroy(() => {

res.clearCookie("connect.sid");

res.redirect(process.env.FRONTEND\_URL);

});

});

} else res.redirect(process.env.FRONTEND\_URL);

});

# Frontend (React)

**🧩 Technologies**

* **React + React Router**
* **Axios** pour communiquer avec le backend
* **CSS personnalisé Harry Potter**
* **Animation 2D** lors du chargement (fichier animation\_2D.html)

**⚙️ Fonctionnalités principales**

* Connexion OAuth2 via un bouton “Se connecter avec Google”.
* Récupération et affichage :
  + des **e-mails** (sujet, date, contenu)
  + des **événements de calendrier** à venir
* Gestion du **logout**
* Thème magique ✨ inspiré de Poudlard

# Infrastructure Docker

**docker-compose.yml**

version: "3.9"

services:

node-backend:

build: ./backend

ports:

- "5000:5000"

env\_file:

- ./backend/.env

depends\_on:

- mongo

react-frontend:

build: ./frontend

ports:

- "3000:3000"

env\_file:

- ./frontend/.env

mongo:

image: mongo

ports:

- "27017:27017"

volumes:

- mongo-data:/data/db

volumes:

mongo-data:

# Authentification Google OAuth2

**Étapes côté Google Cloud**

1. Créer un projet sur console.cloud.google.com.
2. Activer les API :
   * Gmail API
   * Google Calendar API
3. Configurer l’écran de consentement OAuth :
   * Type : *Externe*
   * Ajouter tes utilisateurs testeurs
4. Créer un identifiant OAuth2 :
   * Type : *Client Web*
   * URI autorisée :
     + http://localhost:5000/auth/google/callback
     + http://localhost:3000

# API Gmail et Calendar

**/gmail**

* Méthode : GET
* Authentification requise
* Récupère les 10 derniers e-mails (sujet, date, contenu simplifié)

**/calendar**

* Méthode : GET
* Authentification requise
* Retourne les événements à venir à partir de la date actuelle

# Persistance de session

* Gérée par **express-session**
* Sauvegardée dans **MongoDB** via connect-mongo
* L’utilisateur reste connecté tant que sa session est active

# Tests unitaires

**📁 Structure**

backend/tests/app.test.js

**🧱 Exemple**

jest.mock("connect-mongo", () => {

const session = require("express-session");

class MockStore extends session.Store {

get() {}

set() {}

destroy() {}

}

return { create: () => new MockStore() };

});

jest.mock("mongoose", () => ({ connect: jest.fn() }));

const request = require("supertest");

const app = require("../index");

describe("Hedwige backend - Gmail / Calendar / Logout", () => {

test("GET /me sans connexion renvoie 401", async () => {

const res = await request(app).get("/me");

expect(res.status).toBe(401);

});

test("GET /logout sans connexion redirige vers frontend", async () => {

const res = await request(app).get("/logout");

expect(res.status).toBe(302);

});

});

# Fichiers .env et sécurité

**.env backend**

PORT=5000

SESSION\_SECRET=ton\_secret

GOOGLE\_CLIENT\_ID=xxxx.apps.googleusercontent.com

GOOGLE\_CLIENT\_SECRET=xxxxx

MONGO\_URI=mongodb://mongo:27017/hedwige

FRONTEND\_URL=http://localhost:3000

**.env frontend**

REACT\_APP\_API\_URL=http://localhost:5000

🛡️ Ajoute dans ton .gitignore :

# Ignorer les fichiers sensibles

.env

frontend/.env

backend/.env

# Démarrage du projet

**En local avec Docker :**

docker-compose up --build

Puis ouvre :  
👉 <http://localhost:3000>

# Pistes d’évolution

* Ajout de **Google Drive** ou **Google Tasks**
* Intégration d’un système de **notifications push**
* Historique des connexions
* Interface “Dark Magic” optionnelle 🪄

# Démarrage du projet

**En local avec Docker :**

docker-compose up --build

Puis ouvre :  
👉 <http://localhost:3000>

# Pistes d’évolution

* Ajouter des intégrations supplémentaires :
  + Google Drive
  + Google Tasks
* Ajouter un système de notifications (Web Push)
* Mode “Dark Magic” 🪄
* Support multi-utilisateurs avec rôles