Sommaire

Introduction

Organiz'heure est une application web intuitive et performante conçue pour optimiser la gestion des tâches et des listes au quotidien. Cette solution s’adresse aussi bien aux particuliers qu’aux professionnels cherchant un outil simple et efficace pour organiser leurs activités.

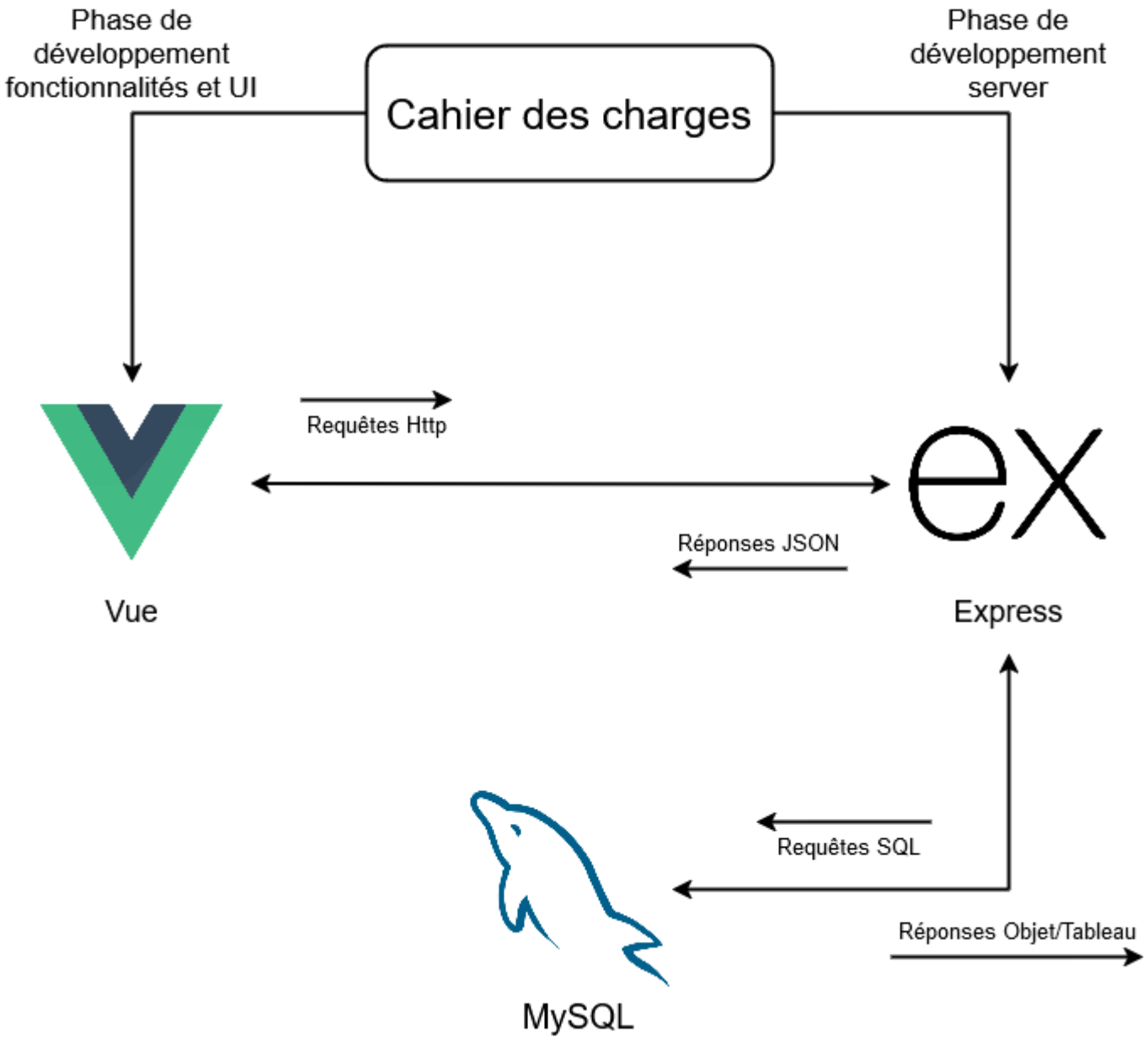
Fonctionnalités principales

- **Création de tâches** : Ajoutez rapidement des tâches avec des descriptions pour garder une trace de ce qui doit être accompli.
- **Gestion des listes** : Organisez vos tâches en listes personnalisées selon vos besoins.
- **Catégories** : Assignez des catégories à vos listes pour une organisation thématique et un accès simplifié.
- **Archivage** : Archivez vos listes terminées pour conserver un historique tout en maintenant une interface claire.

Objectifs

Organiz'heure a pour vocation de simplifier la planification et le suivi des tâches grâce à une interface ergonomique et des outils flexibles. Conçue dans une perspective de facilité d'utilisation, cette application web garantit une expérience utilisateur fluide tout en étant suffisamment robuste pour répondre à des besoins variés.

Schéma d’Architecture Technique



Architecture de la base de données

Dictionnaire de données

Nom	Type	Taille	Description
idUser	INT	10	Id utilisateur
isAdmin	BOOLEAN	/	Est admin ou non
userName	VARCHAR	50	Nom utilisateur
userSurname	VARCHAR	50	Prénom utilisateur
userMail	VARCHAR	50	E-mail utilisateur
hashedPass	VARCHAR	70	Mot de passe utilisateur
idTask	INT	10	Id d'une tâche
labelTask	VARCHAR	50	Libellé d'une tâche
dueTask	DATE	/	Date de la tâche
completionStateTask	BOOLEAN	/	Etat de la tâche
creationTask	DATETIME	/	Time Stamp de création d'une tâche
updateTask	DATETIME	/	Time Stamp de mise à jour d'une tâche
completionTimeTask	DATETIME	/	Time Stamp de complétion d'une tache
idList	INT	10	Id d'une liste
labelList	VARCHAR	50	Label d'une liste
listCreationTime	DATETIME	/	Time Stamp de création d'une liste
listUpdateTime	DATETIME	/	Time Stamp de mise à jour d'une liste
isPersonnal	BOOLEAN	/	Définit si une liste est personnelle ou non
idCategory	INT	10	Id d'une catégorie
labelCategory	VARCHAR	50	Nom d'une catégorie
isAccEnabled	BOOLEAN	/	Compte user est activé ou désactivé
archiveTime	DATETIME	/	Time Stamp d'archivage d'une liste
isArchived	BOOLEAN	/	Est-ce qu'une liste est archivée ou non

Dépendances fonctionnelles

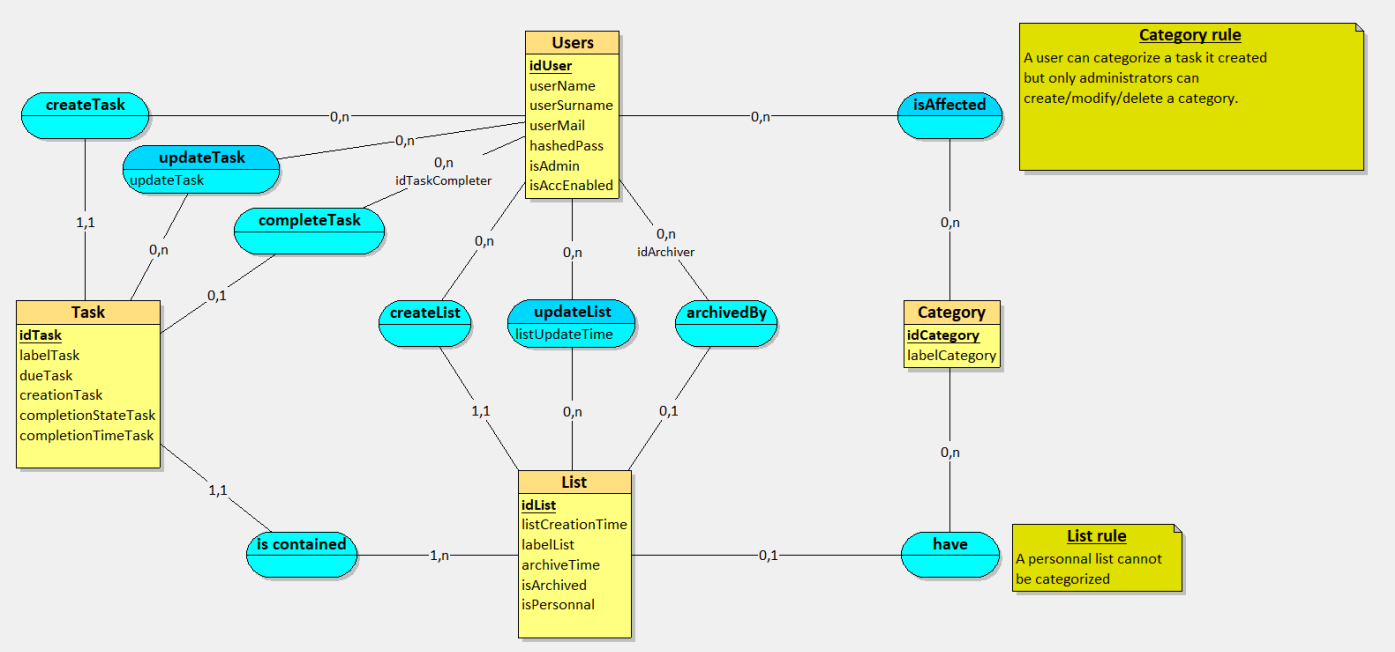
IdUser ? userName, userSurname, userMail, hashedPass, isAdmin, isAccEnabled

IdTask ? labelTask, dueTask, completionStateTask, creationTask, updateTask, completionTimeTask

IdList ? labelList, listCreationTime, listUpdateTime, isPersonnal, isArchived, archiveTime

IdCategory ? labelCategory

Modèle Conceptuel de Données



Modèle Logique de Données

Users = (idUser INT, userName VARCHAR(50), userSurname VARCHAR(50), userMail VARCHAR(50), hashedPass VARCHAR(70), isAdmin LOGICAL, isAccEnabled LOGICAL);

Category = (idCategory INT, labelCategory VARCHAR(50));

List = (idList INT, listCreationTime DATETIME, labelList VARCHAR(50), archiveTime DATETIME, isArchived LOGICAL, isPersonnal LOGICAL, #idArchiver*, #idUser, #idCategory*);

Task = (idTask INT, labelTask VARCHAR(50), dueTask DATE, creationTask DATETIME, completionStateTask LOGICAL, completionTimeTask DATETIME, #idTaskCompleter*, #idList, #idUser);

isAffected = (#idUser, #idCategory);

updateTask = (#idUser, #idTask, updateTask DATETIME);

Code de création SQL

```
CREATE TABLE Users (
  idUser INT AUTO_INCREMENT,
  userName VARCHAR(50) NOT NULL,
  userSurname VARCHAR(50) NOT NULL,
  userMail VARCHAR(50) NOT NULL UNIQUE,
  hashedPass VARCHAR(70) NOT NULL,
  isAdmin BOOLEAN NOT NULL DEFAULT FALSE,
  isAccEnabled BOOLEAN NOT NULL DEFAULT TRUE,
  PRIMARY KEY (idUser)
);

CREATE TABLE Category (
  idCategory INT AUTO_INCREMENT,
  labelCategory VARCHAR(50) NOT NULL UNIQUE,
  PRIMARY KEY (idCategory)
);

CREATE TABLE List (
  idList INT AUTO_INCREMENT,
  labelList VARCHAR(50) NOT NULL,
  listCreationTime DATETIME NOT NULL,
  listUpdateTime DATETIME,
  archiveTime DATETIME,
  isArchived BOOLEAN NOT NULL DEFAULT FALSE,
  isPersonnal BOOLEAN NOT NULL DEFAULT FALSE,
  idArchiver INT,
  idUser INT NOT NULL,
  idCategory INT,
  PRIMARY KEY(idList),
  FOREIGN KEY(idArchiver) REFERENCES Users(idUser),
```

```
  FOREIGN KEY(idUser) REFERENCES Users(idUser),
  FOREIGN KEY(idCategory) REFERENCES Category(idCategory)
);

CREATE TABLE Task (
  idTask INT AUTO_INCREMENT,
  labelTask VARCHAR(50) NOT NULL,
  dueTask DATE NOT NULL,
  completionStateTask BOOLEAN NOT NULL DEFAULT FALSE,
  completionTimeTask DATETIME,
  creationTask DATETIME NOT NULL,
  idTaskCompleter INT,
  idList INT NOT NULL,
  idUser INT NOT NULL,
  PRIMARY KEY(idTask),
  FOREIGN KEY(idTaskCompleter) REFERENCES Users(idUser),
  FOREIGN KEY(idList) REFERENCES List(idList),
  FOREIGN KEY(idUser) REFERENCES Users(idUser)
);

CREATE TABLE isAffected(
  idUser INT,
  idCategory INT,
  PRIMARY KEY(idUser, idCategory),
  FOREIGN KEY(idUser) REFERENCES Users(idUser),
  FOREIGN KEY(idCategory) REFERENCES Category(idCategory)
);

CREATE TABLE updateTask(
  IdUpdateTask INT,
  idUser INT,
  idTask INT,
  updateTask DATETIME,
  PRIMARY KEY(idUpdateTask),
  FOREIGN KEY(idUser) REFERENCES Users(idUser),
  FOREIGN KEY(idTask) REFERENCES Task(idTask)
);


CREATE TABLE updateList(
  IdUpdateList INT,
  idUser INT,
  idList INT,
  listUpdateTime DATETIME,
  PRIMARY KEY(idUpdateList),
  FOREIGN KEY(idUser) REFERENCES Users(idUser),
  FOREIGN KEY(idTask) REFERENCES Task(idTask)
);
```