SAE 502

Migration de données dans un environnement

NOSQL

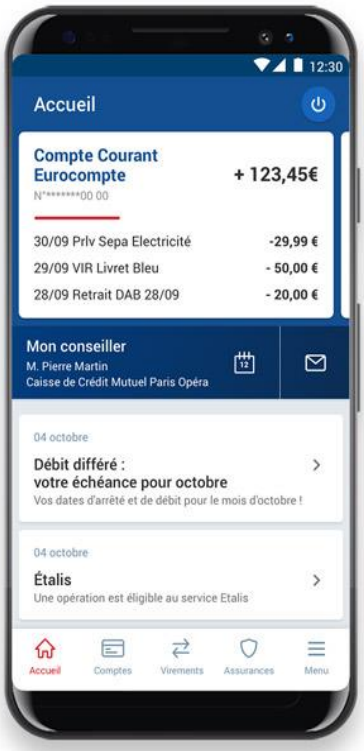
Mamouni Ayoub
Moulin Vincent
BUT 3A VCOD

Table des matières

I- État de l'art

1) Credit Mutuel

Dans notre contexte, nous avons décidé de plusieurs applications que nous allons énumérer au fil des fonctionnalité que nous voulons ajouter à notre application.



Le Crédit Mutuel possède une page d'accueil qui présente une grande partie des informations essentielles que l'utilisateur doit connaître, ainsi que les données que l'on choisit d'afficher.

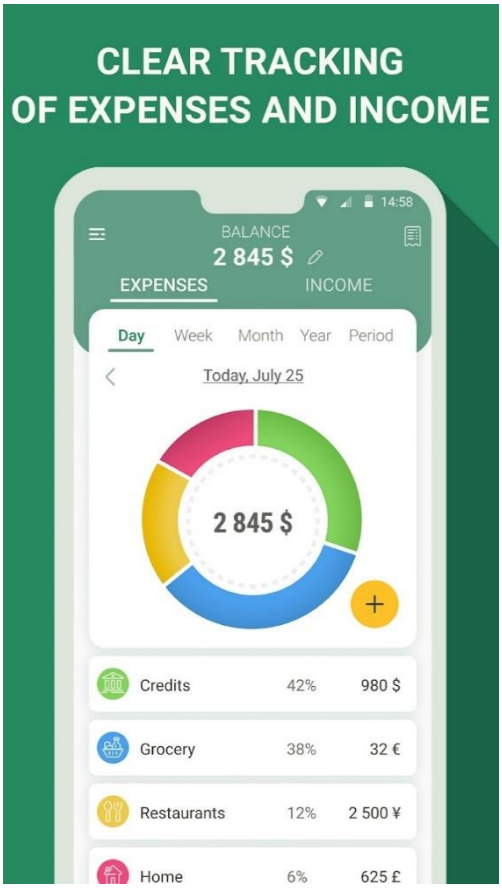
Dans ce cas, on voit d'abord le compte courant de l'utilisateur, avec le solde restant et quelques informations sur les transactions effectuées.

En dessous, des sortes de publicités proposent à l'utilisateur différentes fonctionnalités ou nouveautés susceptibles de l'intéresser.

Plus bas, une barre de navigation permet de se déplacer entre les différentes vues de l'application.

Nous avons décidé de nous baser sur les trois points cités ci-dessus tout en ajoutant des fonctionnalités supplémentaires.

2) Money Manager & Expenses

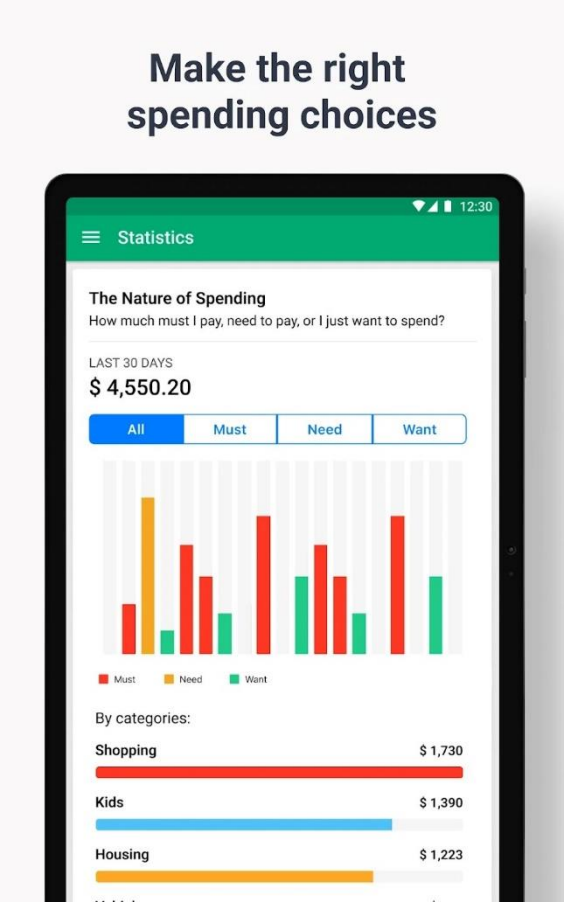


Sur l'application nommée *Money Manager & Expenses*, qui gère les dépenses et les revenus, nous avons décidé de nous inspirer du diagramme circulaire. Celui-ci permettra à l'utilisateur d'avoir une vue globale des différentes dépenses et revenus, ainsi que de comprendre sur quelles catégories il a le plus dépensé ou perçu de revenus.

Dans la partie inférieure, des informations plus détaillées seront affichées, notamment le montant total par catégorie et le pourcentage correspondant.

Une grande partie des vues sont largement inspirées des fonctionnalités de l'application mentionnée ci-dessus. Cependant, nous avons décidé d'ajouter une grande quantité de détails qui ne sont pas forcément présents dans l'application d'origine.

3) Wallet



Pour les sections où l'utilisateur souhaite examiner plus en détail les dépenses d'une catégorie spécifique, nous avons décidé de nous inspirer du diagramme en barres. Celui-ci permettra à l'utilisateur de visualiser les dépenses effectuées sur une période donnée, comme un mois, un trimestre, ou toute autre durée choisie.

Bien sûr, les détails des dépenses ainsi que les enseignes où elles ont été effectuées seront affichés plus bas, comme nous pouvons le voir sur l'image ci-dessus.

4) Pourquoi le NoSQL

Le choix de NoSQL, pour ce projet de gestion de budget s'explique par plusieurs avantages liés à la nature des données et aux besoins d'évolutivité. Les bases de données NoSQL sont idéales pour gérer des données non structurées ou semi-structurées, comme celles des opérations de budget, qui peuvent varier en termes de champs et de types d'information Contrairement aux bases de données relationnelle, qui imposent une structure rigide. NoSQL permet de stocker ces informations de manière plus flexible, ce qui facilite leur évolution au fur et à mesure des besoins de l'application.

En outre, NoSQL, et notamment MongoDB, permet de gérer des volumes de données très importants et de les distribuer efficacement sur plusieurs serveurs. Cela est particulièrement utile dans le contexte dans notre cas d'application de gestion de budget où les utilisateurs génèrent constamment des opérations, et où il est essentiel de garantir des performances élevées pour des requêtes complexes ou des analyses en temps réel. Notamment Grâce à sa capacité à répartir les données sur différents serveurs.

II- Collection

1) Budget

Budget est la collection principal de notre application celle-ci est appeler sur quasiment toute les pages nous avons choisi de ne pas inclure l'utilisateurs dedans

```
{
    "_id": ObjectId(),
    "nom": "String",
    "categorie": {
        "logo": "String",
        "nom": "String",
        "sous_categorie": [
            {
                "logo": "String",
                "nom": "String"
            }
        ]
    },
    "montant": "int",  // Montant de l'opération
    "date": "date",
    "type": "String",  // Type de paiement
     "revenue" : "bool",  // false dépense true revenue
}
```

2) Catégorie

Catégorie est la collection principalement utiliser pour avoir un référentiel de toutes le catégorie définie par la banque et de permettre à l'utilisateurs de définir des seuil sur chaque catégorie existante

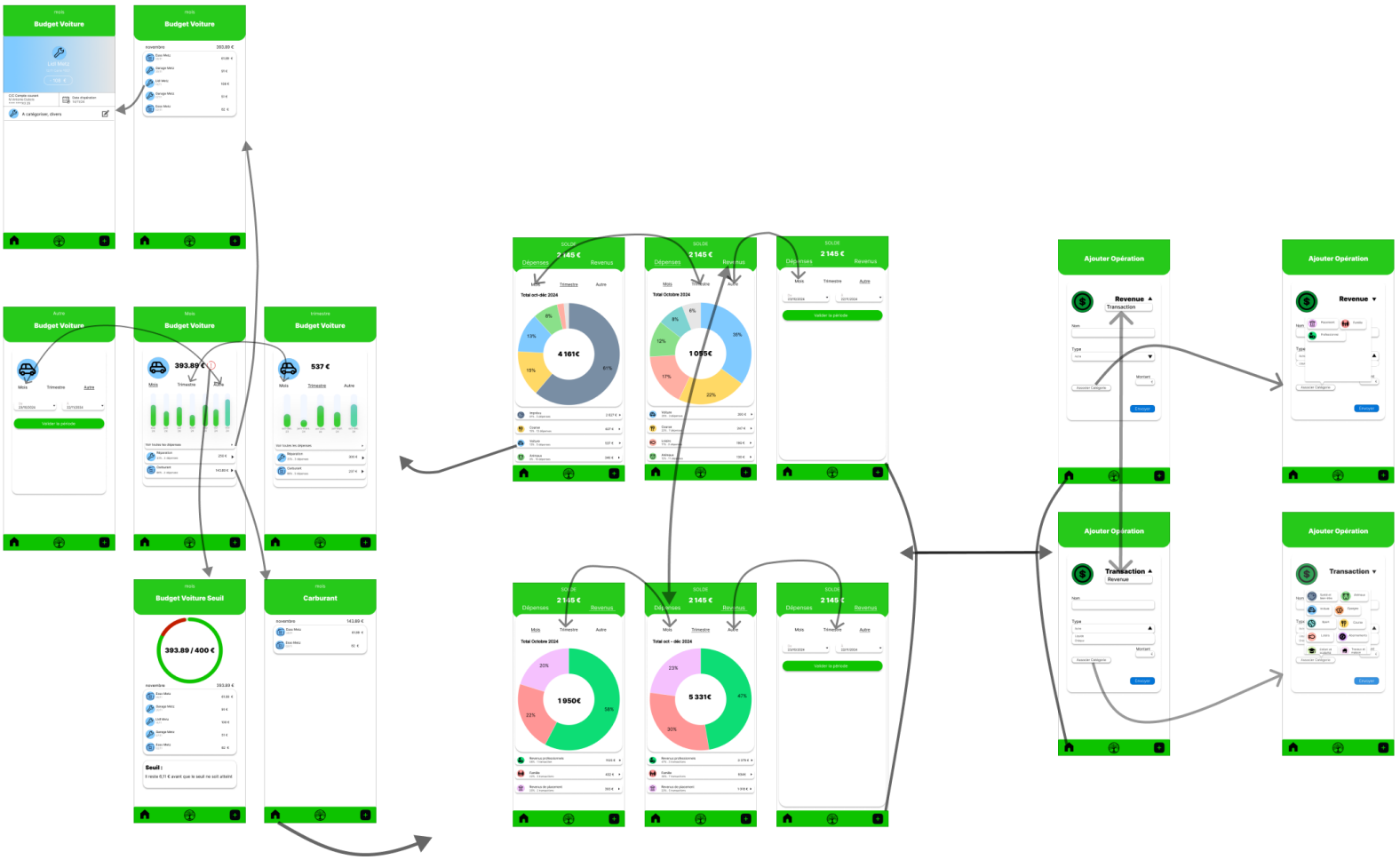
```
{
    "_id": ObjectId(),
    "nom": "String",
    "logo" : "string",
    "revenue" : false,
    "seuil": {
         "montant" : "int"
         "type" : "string" //mois trimestre année
     }
}
```

3) Utilisateurs

Collection utilisateur utiliser que sur une page, n'est pas utile à notre logique de notre application mais nécessaire pour l'application de banque dans son objectif de passer dans un environnement noSql (formulaire de connexion, gestion des comptes, etc..)

```
{
    "_id": ObjectId(),
    "nom": "String",
    "prenom": "String",
    "comptes": [
        {
            "nom": "String",
            "num": "String"
        }
    ],
    "cartes": [
        {
            "nom": "String",
            "num": "String"
        }
    ]
}
```

III- Schéma Fonctionnel

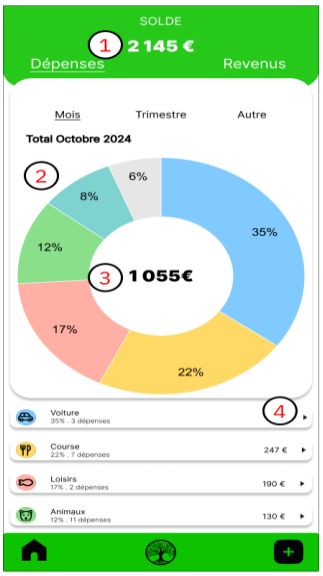


- Navbar
  - Maison : permet de revenir à l'accueil
  - Arbre : permet de revenir sur l'application bancaire
  - Plus : permet d'ajouter des transaction hors système bancaire
- Catégorie (au clic sur un budget)
  - Vues des sous-catégorie ainsi que la possibilité de voir le detail des transaction
  - Possibilité de changer une transaction de catégorie
  - Définition de seuil

IV- Vues

1) Accueil

a) Depense




```
#resultat attendue
[ { solde: 798.81 } ]

#requête
db.budget.aggregate([
    {
        $group: {
            _id: null,
            totalDepenses: { $sum: { $cond: [{ $eq: ["$revenue", false] }, "$montant", 0] } },
            totalRevenus: { $sum: { $cond: [{ $eq: ["$revenue", true] }, "$montant", 0] } }
        }
    },
    {
        $project: { _id: 0, solde: { $subtract: ["$totalRevenus", "$totalDepenses"] } }
    }
]);
```

1

```
#resultat
[{ _id: 'Santé', total: 183.2 }, { _id: 'Maison', total: 309 }]

#requête
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            date: {
                $gte: new Date(new Date().getFullYear(), new Date().getMonth(), 1), // Debut mois
                $lt: new Date(new Date().getFullYear(), new Date().getMonth() + 1, 1) // Debut mois suivant
            }
        }
    },
    {
        $group: {_id: "$categorie.nom", total: { $sum: "$montant" } }
    }
]);
```

2

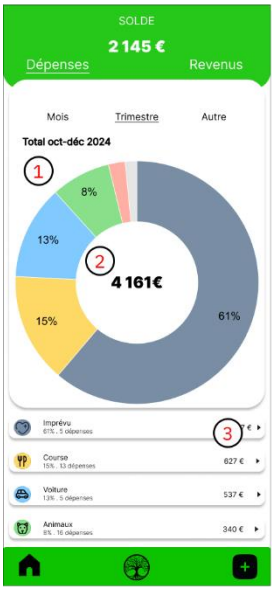
```
#resultat
[ { _id: null, total: 801.19 } ]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            date: {
                $gte: new Date(new Date().getFullYear(), new Date().getMonth(), 1),
                $lt: new Date(new Date().getFullYear(), new Date().getMonth() + 1, 1)
            }
        }
    },
    {
        $group: {_id: null, total: { $sum: "$montant" } }
    }
]);
```

3

```
#resultat
[ { montant: 60.99, nombreDepenses: 2, categorie: 'Loisirs',
  logo: '/logos/loisirs.png' },
  { montant: 115.5, nombreDepenses: 2, categorie: 'Voiture', ,
  logo: '/logos/voiture.png' } ]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            date: {
                $gte: new Date(new Date().getFullYear(), new Date().getMonth(), 1),
                $lt: new Date(new Date().getFullYear(), new Date().getMonth() + 1, 1)
            }
        }
    },
    {
        $group: {
            _id: { nom: "$categorie.nom", logo: "$categorie.logo" },
            montant: { $sum: "$montant" },  nombreDepenses: { $sum: 1 }
        }
    },
    {
        $project: {  _id: 0, categorie: "$_id.nom", logo: "$_id.logo", montant: 1, nombreDepenses: 1 }
    }
]);
```

4

Cette vue représente les dépenses que l'utilisateur à effectuer tout au long du mois d'octobre.




```
#resultat
[{ _id: 'Santé', total: 383.2 },
{ _id: 'Loisirs', total: 709 }]

#requête
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            date: {
                $gte: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3, 1), // calcul trimestre
                $lt: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3 + 3, 0) // 0 = dernier jour mois
            }
        }
    },
    {
        $group: { _id: "$categorie.nom", total: { $sum: "$montant" } }
    }
]);
```

1

```
#resultat
[ { _id: null, total: 431.19 } ]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            date: {
                $gte: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3, 1),
                $lt: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3 + 3, 0)
            }
        }
    },
    {
        $group: { _id: null, total: { $sum: "$montant" } }
    }
]);
```

2

```
#resultat
[ { montant: 87.99, nombreDepenses: 2, categorie: 'Loisirs', logo: '/logos/loisirs.png'},
{ montant: 345.8, nombreDepenses: 2, categorie: 'Animaux', logo: '/logos/animaux.png' } ]

#requete

db.budget.aggregate([
    {
        $match: {
            revenue: false,
            date: {
                $gte: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3, 1),
                $lt: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3 + 3, 0)
            }
        }
    },
    {
        $group: {
            _id: { nom: "$categorie.nom", logo: "$categorie.logo" },
            montant: { $sum: "$montant" }, nombreDepenses: { $sum: 1 }
        }
    },
    {
        $project: { _id: 0, categorie: "$_id.nom", logo: "$_id.logo", montant: 1, nombreDepenses: 1 }
    }
]);
```

3

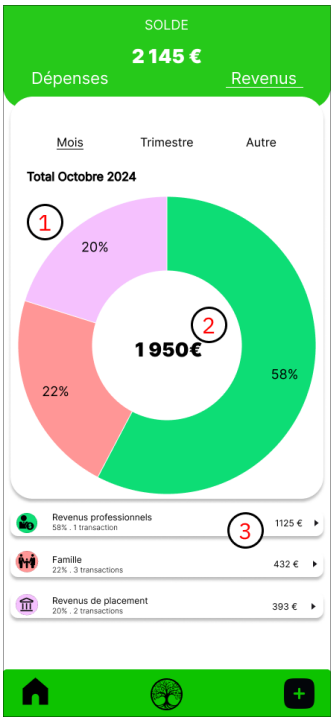
Cette vue représente les dépenses que l’utilisateur a effectuées tout au long du trimestre.

1

```
#resultat
[{ montant: 60.99, nombreDepenses: 2, categorie: 'Loisirs' },
{ montant: 115.5, nombreDepenses: 2, categorie: 'Voiture' } ]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            date: { $gte: $dateDebut, $lt: $dateFin }
        }
    },
    {
        $group: {_id: "$categorie.nom", montant: { $sum: "$montant" }, nombreDepenses: { $sum: 1 } }
    },
    {
        $project: {_id: 0, categorie: "$_id", montant: 1, nombreDepenses: 1 }
    }
]);
```

Cette vue permet à l'utilisateur de choisir une période et de voir les dépenses effectuer lors de cette période.

b) Revenues




```
#resultat
[{ _id: 'Famille', total: 183.2 },
{ _id: 'Revenues Professionnel', total: 309 }]

#requête
db.budget.aggregate([
    {
        $match: {
            revenue: true,
            date: {
                $gte: new Date(new Date().getFullYear(), new Date().getMonth(), 1),
                $lt: new Date(new Date().getFullYear(), new Date().getMonth() + 1, 1)
            }
        }
    },
    {
        $group: {_id: "$categorie.nom", total: { $sum: "$montant" } }
    }
]);
```

1

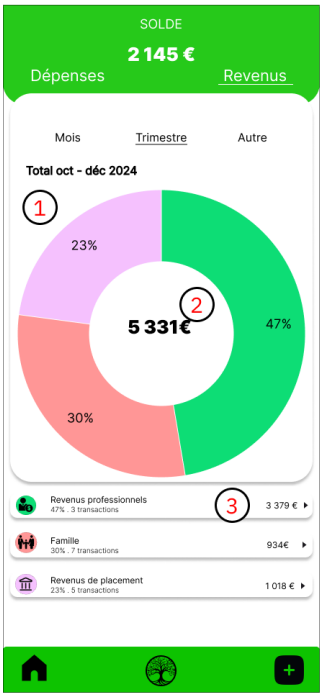
```
#resultat
[ { _id: null, total: 801.19 } ]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: true,
            date: {
                $gte: new Date(new Date().getFullYear(), new Date().getMonth(), 1),
                $lt: new Date(new Date().getFullYear(), new Date().getMonth() + 1, 1)
            }
        }
    },
    {
        $group: {_id: null, total: { $sum: "$montant" } }
    }
]);
```

2

```
#resultat
[{ montant: 60.99, transaction: 2, categorie: 'Famille',logo: '/logos/famille.png' }]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: true,
            date: {
                $gte: new Date(new Date().getFullYear(), new Date().getMonth(), 1),
                $lt: new Date(new Date().getFullYear(), new Date().getMonth() + 1, 1)
            }
        }
    },
    {
        $group: {_id: { nom: "$categorie.nom", logo: "$categorie.logo" },
                 montant: { $sum: "$montant" }, transaction: { $sum: 1 } }
    },
    {
        $project: {_id: 0, categorie: "$_id.nom", logo: "$_id.logo", montant: 1, transaction: 1 }
    }
]);
```

3

Cette vue représente les revenues de l’utilisateur tout au long du mois d’octobre.




```
#resultat
[{ _id: 'Famille', total: 383.2 }]

#requête
db.budget.aggregate([
    {
        $match: {
            revenue: true,
            date: {
                $gte: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3, 1),
                $lt: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3 + 3, 0)
            }
        }
    },
    {
        $group: {_id: "$categorie.nom", total: { $sum: "$montant" } }
    }
]);
```

1

```
#resultat
[ { _id: null, total: 431.19 } ]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: true,
            date: {
                $gte: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3, 1),
                $lt: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3 + 3, 0)
            }
        }
    },
    {
        $group: {_id: null, total: { $sum: "$montant" } }
    }
]);
```

2

```
#resultat
[ { montant: 87.99, transaction: 2, categorie: 'Revenus de placement', logo: '/logos/placement.png' },
{ montant: 345.8, transaction: 2, categorie: 'Famille', logo: '/logos/famille.png' } ]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: true,
            date: {
                $gte: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3, 1),
                $lt: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3 + 3, 0)

            }
        }
    },
    {
        $group: {_id: { nom: "$categorie.nom", logo: "$categorie.logo" },
   montant: { $sum: "$montant" }, transaction: { $sum: 1 } }
    },
    {
        $project: {_id: 0, categorie: "$_id.nom", logo: "$_id.logo", montant: 1, transaction: 1 }
    }
]);
```

3

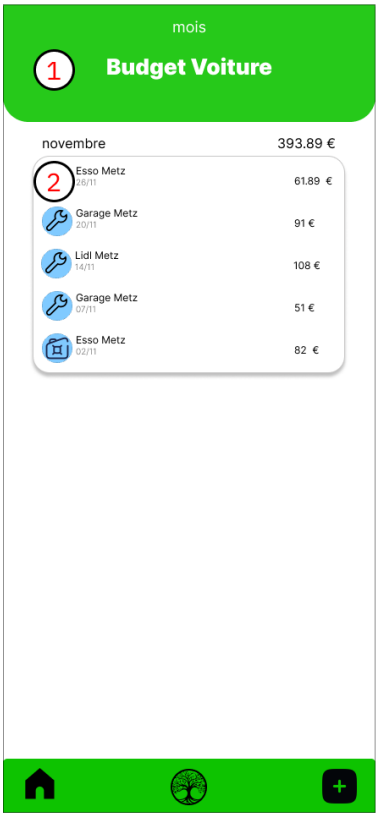
Cette vue représente les revenues de l’utilisateur tout au long du trimestre.


```
#resultat
[{ montant: 60.99, transaction: 2, categorie: 'Revenus professionnel' },
{ montant: 115.5, transaction: 3, categorie: 'Famille' } ]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: true,
            date: { $gte: $dateDebut, $lt: $dateFin }
        }
    },
    {
        $group: {_id: "$categorie.nom", montant: { $sum: "$montant" }, transaction: { $sum: 1 } }
    },
    {
        $project: {_id: 0, categorie: "$_id", montant: 1, transaction: 1 }
    }
]);
```

1

Cette vue permet à l'utilisateur de choisir une période et de voir les revenues effectuées lors de cette période.

2) Voiture détail



1

```
#resultat
[ { total: 115.5, mois: 12 } ]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            "categorie.nom": "Voiture",
            date: {
                $gte: new Date(new Date().getFullYear(), new Date().getMonth(), 1),
                $lt: new Date(new Date().getFullYear(), new Date().getMonth() + 1, 1)
            }
        }
    },
    {
        $group: { _id: null, total: { $sum: "$montant" } }
    },
    {
        $project: { _id: 0, mois: { $month: new Date() },  total: 1 }
    }
]);
```

2

```
#resultat
[{ nom: 'Norauto Metz', montant: 45.5,
    date: ISODate('2024-12-17T19:15:49.684Z'), logo: '/logos/entretien.png'},
  { nom: 'Station Total', montant: 70,
  date: ISODate('2024-12-17T19:15:49.684Z'), logo: '/logos/essence.png' }]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            date: {
                $gte: new Date(new Date().getFullYear(), new Date().getMonth(), 1),
                $lt: new Date(new Date().getFullYear(), new Date().getMonth() + 1, 1)
            },
            "categorie.nom": "Voiture"
        }
    },
    {
        $project: { _id: 0, nom: 1, montant: 1, date: 1, logo: "$categorie.sous_categorie.logo" }
    }
]);
```

Cette vue représente les dépenses de la catégorie voiture effectuer par l’utilisateur lors du mois de novembre.

1

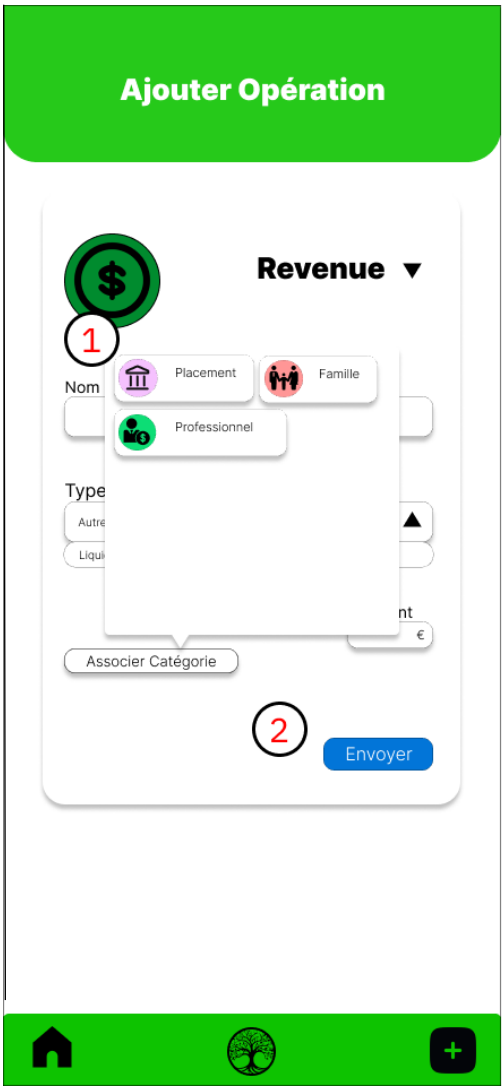
```
#resultat
[ { total: 70.5, mois: 12 } ]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            "categorie.sous_categorie.nom": "Essence",
            date: {
                $gte: new Date(new Date().getFullYear(), new Date().getMonth(), 1),
                $lt: new Date(new Date().getFullYear(), new Date().getMonth() + 1, 1)
            }
        }
    },
    {
        $group: { _id: null, total: { $sum: "$montant" } }
    },
    {
        $project: { _id: 0, mois: { $month: new Date() },  total: 1 }
    }
]);
```

2

```
#resultat
[{ nom: 'Station Total', montant: 70, date: ISODate('2024-12-17T19:15:49.684Z') }]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            date: {
                $gte: new Date(new Date().getFullYear(), new Date().getMonth(), 1),
                $lt: new Date(new Date().getFullYear(), new Date().getMonth() + 1, 1)
            },
            "categorie.sous_categorie.nom": "Essence"
        }
    },
    {
        $project: { _id: 0, montant: 1, nom: 1, date: 1 }
    }
]);
```

Cette vue représente les dépenses de la catégorie carburant effectuer par l'utilisateur lors du mois de novembre.

3) Opération



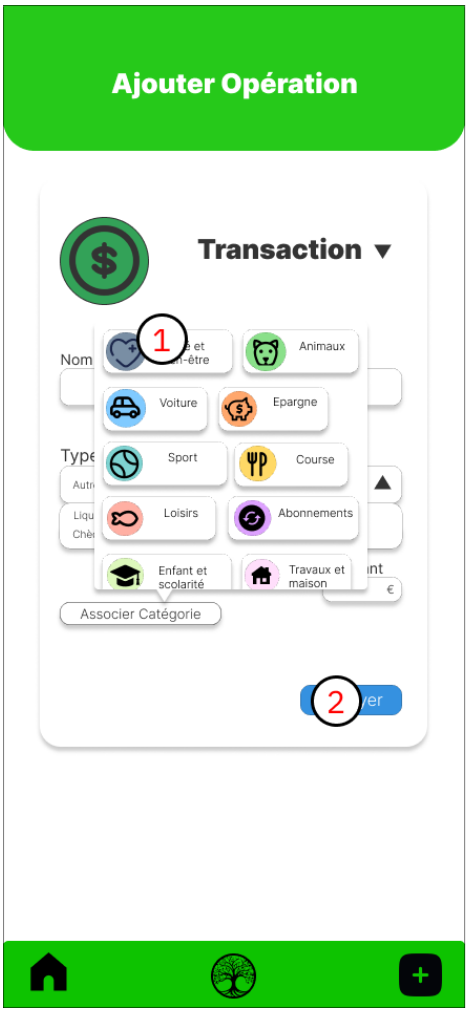
1

```
#resultat
[ { nom: 'Placement', logo: 'logos/placement' } ]

#requete
db.categorie.find( {revenue: true}, { _id: 0, nom: 1, logo: 1 } );
```

2

```
#requete
db.budget.insertOne({
    "nom": $nom, //champ formulaire
    "categorie": {
        "logo": "/logos/famille.png", //choix user
        "nom": "Famille",
    },
    "montant": $montant, //form
    "date": new Date(),
    "type": $type, //form
    "revenue": true
});
```

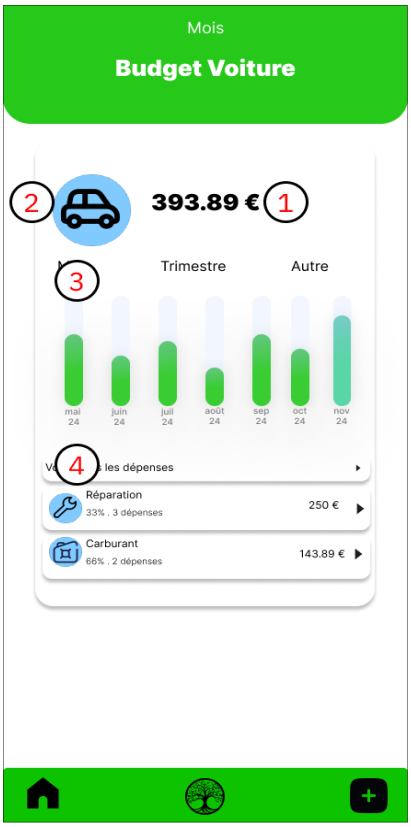
Cette vue permet à l'utilisateur de choisir une catégorie selon le revenu choisie.




```
#resultat
[ { nom: 'Santé', logo: 'logos/sante' } ]

#requete
db.categorie.find({revenue: false},{_id: 0, nom: 1, logo: 1 } );

#requete
db.budget.insertOne({
    "nom": $nom, //champ formulaire
    "categorie": {
        "logo": "/logos/loisirs.png", //choix user
        "nom": "Loisirs",
    },
    "montant": $montant, //form
    "date": new Date(),
    "type": $type, //form
    "revenue": false
});
```

Cette vue permet à l'utilisateur de choisir une catégorie selon la dépense choisie

4) Voiture




```
#resultat
[ { montant: 115.5 } ]

#requête
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            "categorie.nom": "Voiture",
            date: {
                $gte: new Date(new Date().getFullYear(), new Date().getMonth(), 1),
                $lt: new Date(new Date().getFullYear(), new Date().getMonth() + 1, 1)
            }
        }
    },
    {
        $group: { _id:  null, montant: { $sum: "$montant" } },
    {
        $project: { _id: 0, montant: 1 }
    },
]);
```

1

```
#resultat
[ { logo: '/logos/voiture.png' } ]

#requete
db.categorie.find( { "nom": "Voiture" }, { "logo": 1, "_id": 0 } );
```

2

```
#resultat
[ { montant: 70 }, { montant: 45.5 }, { montant: 115.5 } ]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            "categorie.nom": "Voiture",
            date: {
                $gte: new Date(new Date().getFullYear(), new Date().getMonth() - 6, 1),
                $lt: new Date(new Date().getFullYear(), new Date().getMonth() + 1, 1)
            }
        }
    },
    {
        $group: { _id: { mois: { $month: "$date" } }, montant: { $sum: "$montant" } }
    },
    {
        $sort: { "_id.mois": 1 }
    },
    {
        $project: {  _id: 0, montant: 1 }
    }
]);
```

3

```
#resultat
[{montant: 70, nombreDepenses: 1, categorie: [ 'Essence' ], logo: [ '/logos/essence.png' ]},
 {montant: 45.5, nombreDepenses: 1, categorie: [ 'Accessoires' ], logo: [ '/logos/entretien.png' ]}]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: false,
   "categorie.nom" : "Voiture",
            date: {
    $gte: new Date(new Date().getFullYear(), new Date().getMonth(), 1),
                $lt: new Date(new Date().getFullYear(), new Date().getMonth()+ 1, 0)
            }
        }
    },
    {
        $group: {
            _id: { nom: "$categorie.sous_categorie.nom", logo: "$categorie.sous_categorie.logo" },
            montant: { $sum: "$montant" }, nombreDepenses: { $sum: 1 }
        }
    },
    {
        $project: { _id: 0, categorie: "$_id.nom", logo: "$_id.logo", montant: 1, nombreDepenses: 1 }
    }
]);
```

4

Cette vue permet à l'utilisateur d'avoir une vue sur les dépenses effectuer sur plusieurs moi et de choisir quel mois afficher.

1

```
#resultat
[ { montant: 345.5 } ]

#requête
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            "categorie.nom": "Voiture",
            date: {
                $gte: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3, 1),
                $lt: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3 + 3, 1)
            }
        }
    },
    {
        $group: { _id: null, montant: { $sum: "$montant" }}
    },
    {
        $project: { _id: 0, montant: 1 }
    }
]);
```

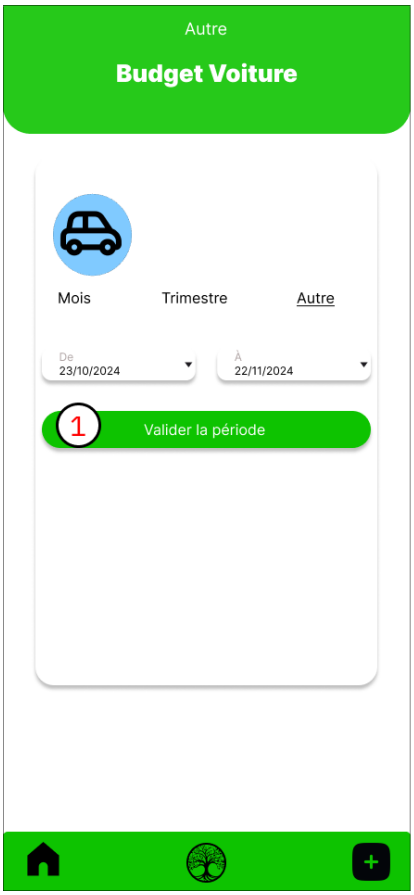
2

```
#resultat
[{ montant: 45.5, mois: 10, year: 2024 }, { montant: 115.5, mois: 12, year: 2024 }]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            "categorie.nom": "Voiture",
            date: {
                $gte: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3, 1),
                $lt: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3 + 3, 1)
            }
        }
    },
    {
        $group: {
            _id: { mois: { $month: "$date" }, annee: { $year: "$date" } },
            montant: { $sum: "$montant" }
        }
    },
    {
        $sort: { "_id.mois": 1 }
    },
    {
        $project: { _id: 0, mois: "$_id.mois", annee: "$_id.annee", montant: 1 }
    }
]);
```

3

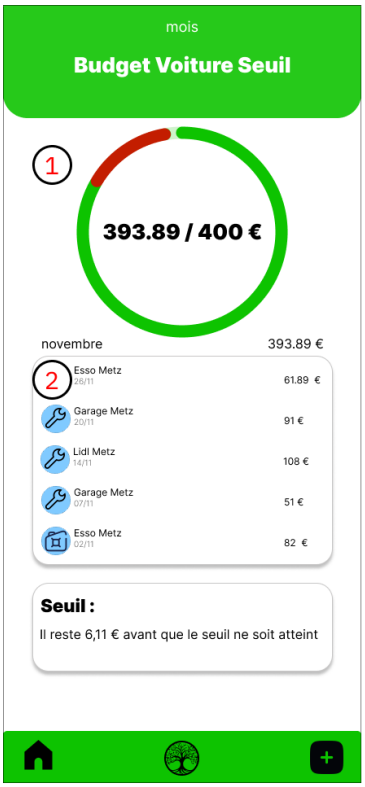
```
#resultat
[{montant: 780, nombreDepenses: 14, categorie: [ 'Essence' ], logo: [ '/logos/essence.png' ]},
 {montant: 64.5, nombreDepenses: 5, categorie: [ 'Entretien' ], logo: [ '/logos/entretien.png' ]}]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: false,
   "categorie.nom" : "Voiture",
            date: {
    $gte: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3, 1),
                $lt: new Date(new Date().getFullYear(), Math.floor(new Date().getMonth() / 3) * 3 + 3, 1)
            }
        }
    },
    {
        $group: {
            _id: { nom: "$categorie.sous_categorie.nom", logo: "$categorie.sous_categorie.logo" },
            montant: { $sum: "$montant" }, nombreDepenses: { $sum: 1 }
        }
    },
    {
        $project: { _id: 0, categorie: "$_id.nom", logo: "$_id.logo", montant: 1, nombreDepenses: 1 }
    }
]);
```

Cette vue permet à l'utilisateur d'avoir une vue sur les dépenses du budget voiture effectuer sur plusieurs trimestres et de choisir quel trimestre afficher.




```
#resultat
[{montant: 20, nombreDepenses: 1, categorie: [ 'Essence' ], logo: [ '/logos/essence.png' ]},
 {montant: 65, nombreDepenses: 2, categorie: [ 'Entretien' ], logo: [ '/logos/entretien.png' ]}]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            "categorie.nom" : "Voiture",
            date: {
                $gte: $date_debut, //choix utilisateur
                $lt: $date_fin
            }
        }
    },
    {
        $group: {
            _id: { nom: "$categorie.sous_categorie.nom", logo: "$categorie.sous_categorie.logo" },
            montant: { $sum: "$montant" }, nombreDepenses: { $sum: 1 }
        }
    },
    {
        $project: { _id: 0, categorie: "$_id.nom", logo: "$_id.logo", montant: 1, nombreDepenses: 1 }
    }
]);
```

Cette vue permet à l'utilisateur de choisir d'afficher les dépenses du budget voiture sur une période sélectionner.



1

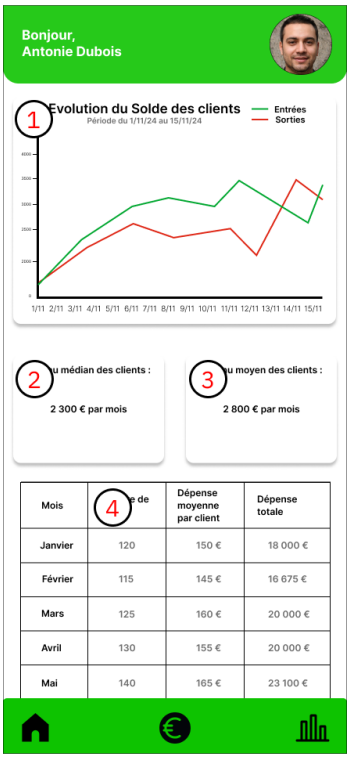
```
#resultat
[ { seuil: { montant: 1000 } } ]

[ { montant: 115.5 } ]

#requete
db.categories.find({ nom: "Voiture", "seuil.type": "mois" },{ _id: 0, "seuil.montant": 1 });

db.budget.aggregate([{
        $match: {
            revenue: false,
            "categorie.nom": "Voiture",
            date: {
                $gte: new Date(new Date().getFullYear(), new Date().getMonth(), 1),
                $lt: new Date(new Date().getFullYear(), new Date().getMonth() + 1, 1)
            }
        }},
    {$group: {_id: null, montant: { $sum: "$montant" }}},
    {$project: {_id: 0, montant: 1}}
]);
```

2

```
#resultat
[{ nom: 'Norauto Metz', montant: 45.5,
    date: ISODate('2024-12-17T19:15:49.684Z'), [logo: '/logos/entretien.png']},
  { nom: 'Station Total', montant: 70,
  date: ISODate('2024-12-17T19:15:49.684Z'), [logo: '/logos/essence.png' ]}]

#requete
db.budget.aggregate([
    {
        $match: {
            revenue: false,
            date: {
                $gte: new Date(new Date().getFullYear(), new Date().getMonth(), 1),
                $lt: new Date(new Date().getFullYear(), new Date().getMonth() + 1, 1)
            },
            "categorie.nom": "Voiture"
        }
    },
    {
        $project: { _id: 0, nom: 1, montant: 1, date: 1, logo: "$categorie.sous_categorie.logo"}
    }
]);
```

Cette vue permet à l’utilisateur d’afficher le seuil du budget voiture.

5) Statistiques



1

```
#resultat
[{ revenus: 0, depenses: 70, jour: 16 },
  { revenus: 0, depenses: 45.5, jour: 17 },
  { revenus: 1600, depenses: 801.19, jour: 18 }]

#requete
db.budget.aggregate([
    {
        $match: {
            date: {
                $gte: new Date(new Date().setDate(new Date().getDate() - 15)),
                $lte: new Date()
            }
        }
    },
    {
        $group: {
            _id: { jour: { $dayOfMonth: "$date" } },
            revenus: { $sum: { $cond: [{ $eq: ["$revenue", true] }, "$montant", 0] } },
            depenses: { $sum: { $cond: [{ $eq: ["$revenue", false] }, "$montant", 0] } }
        }
    },
    { $sort: { "_id.jour": 1 } },
    {
        $project: { _id: 0, jour: "$_id.jour", revenus: 1, depenses: 1 }
    }
]);
```

2

```
#resultat
[ { _id: null, medianMontant: 70 } ]

#requete
db.budget.aggregate([
    {
        $group: {
            _id: null,
            medianMontant: {
                $median: { input: "$montant", method: "approximate" }
            }
        }
    }
]);
```

3

```
#resultat
[ { revenuMoyen: 800 } ]

#requete
db.budget.aggregate([
    {
        $match: { revenue: true }
    },
    {
        $group: {_id: null, revenuMoyen: { $avg: "$montant" }}
 },
    {
        $project: { _id: 0, revenuMoyen: 1}
    }
]);
```

4

```
#resultat
[{ nombreTransaction: 12, depenseTotale: 916.69, mois: 12, depenseMoyenne: 76.39 } ]

#requete
db.budget.aggregate([
    {
        $match: { revenue: false }
    },
    {
        $group: {
            _id: { mois: { $month: "$date" } },
            nombreTransaction: { $sum: 1 },
            depenseTotale: { $sum: "$montant" },
            depenseMoyenne: { $avg: "$montant" }
        }
    },
    { $sort: { "_id.mois": 1 } },
    {
        $project: {
            _id: 0,
            mois: "$_id.mois",
            nombreTransaction: 1,
            depenseMoyenne: { $round: ["$depenseMoyenne", 2] },
            depenseTotale: 1
        }
    }
]);
```

Cette vue permet d’afficher les données de tous les utilisateurs.

V- Conclusion

Dans ce projet, nous avons créé une structure de données adaptée à la gestion de budget personnel en utilisant une base de données NoSQL. Nous avons opté pour une approche flexible, permettant de gérer des informations complexes sur les opérations financières, les catégories de dépenses, et les utilisateurs. La relation entre les opérations de budget, les catégories et les utilisateurs est clairement définie, ce qui permet de suivre les flux financiers tout en offrant une grande souplesse dans l’ajout de nouvelles fonctionnalités à l'avenir.

Pour une application de gestion de budget, cela permettrait aux utilisateurs de visualiser et d'analyser facilement leurs dépenses par catégorie, suivre leur solde global et gérer leurs finances personnelles de manière simple et intuitive.

La mise en place de ce système dans cette banque nécessite une intégration fluide avec les systèmes existants, en assurant une collecte et un traitement sécurisés des données utilisateurs. En s’appuyant sur MongoDB, la banque pourrait rapidement faire évoluer l’application en fonction des besoins des utilisateurs tout en garantissant une haute disponibilité des données.

Figma Démo

Figma Vue

Figma Requette