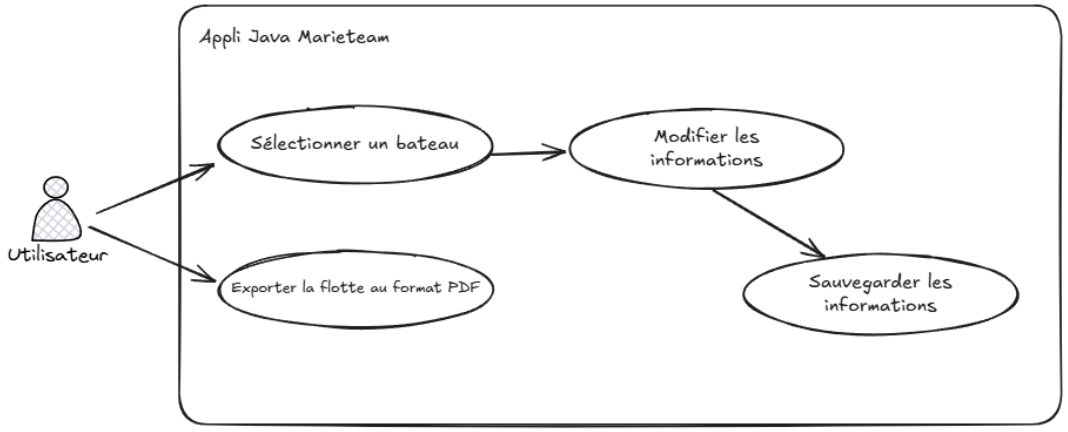
Marieteam Java

Contexte

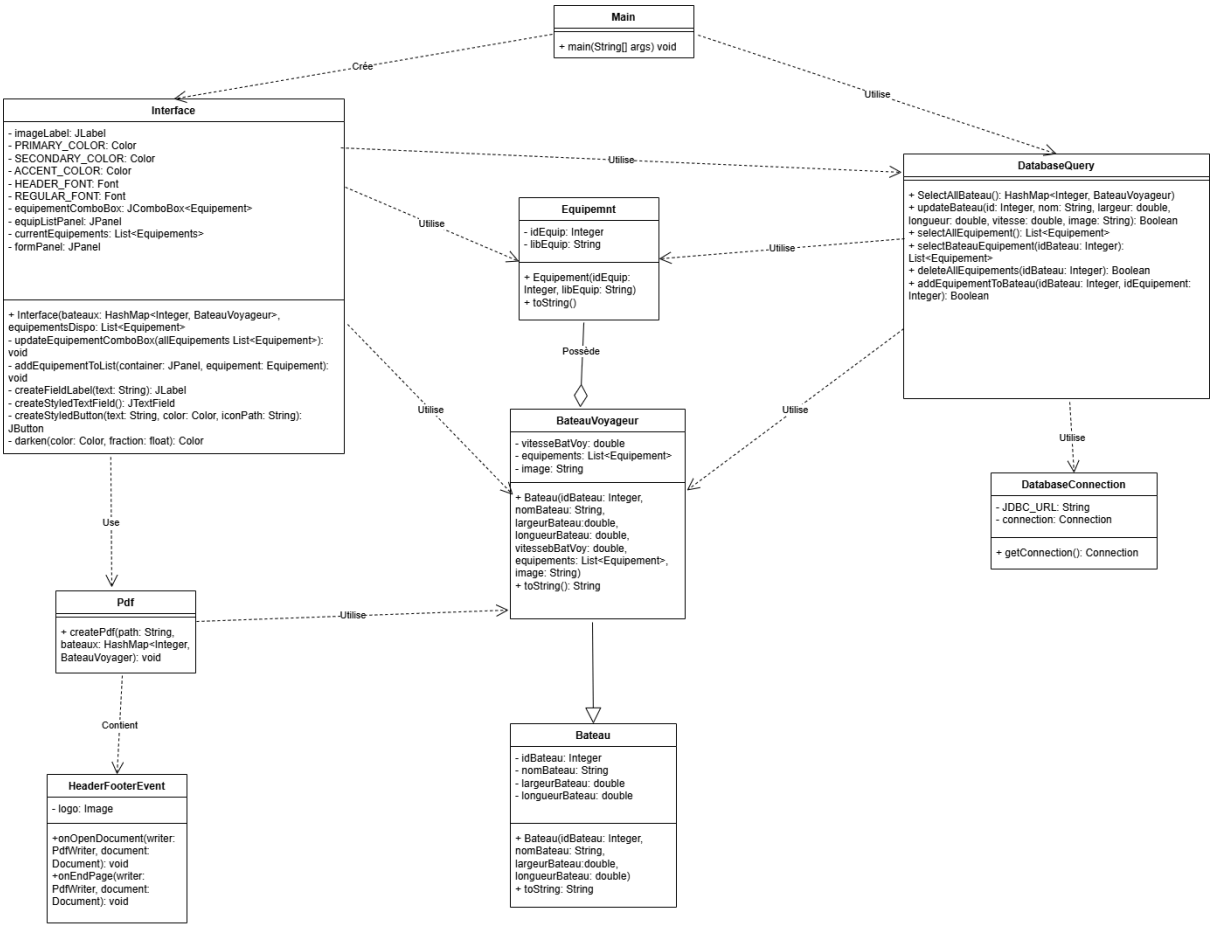
Dans le cadre de l'épreuve E6, il fallait réaliser un client lourd permettant l'édition de brochure récapitulant la flotte de bateaux de Marieteam. Il fallait aussi permettre la modification par les utilisateurs, des informations d'un bateau. Le projet devait utiliser la même base de données que l'application web, de même que fournir une interface graphique pour l'utilisateur.

I. Les technologies

Le projet est développé en Java. Il utilise les bibliothèques ItextPDF pour la création de PDF et Swing pour les interfaces, aussi Junit pour les test unitaires, aussi postgresql pour faire les requêtes à la bdd



II. Diagramme de classe



III. Interface

L'interface est géré dans la classe Interface. La librairie Swing est utilisé.

Voici comment on la déclare dans la méthode Main:

```
SwingUtilities.invokeLater(new Runnable() {
        @Override
        public void run() {
            new Interface(finalBateaux, equipementDispo);  // Création de la f
enêtre graphique
        }
    });
```

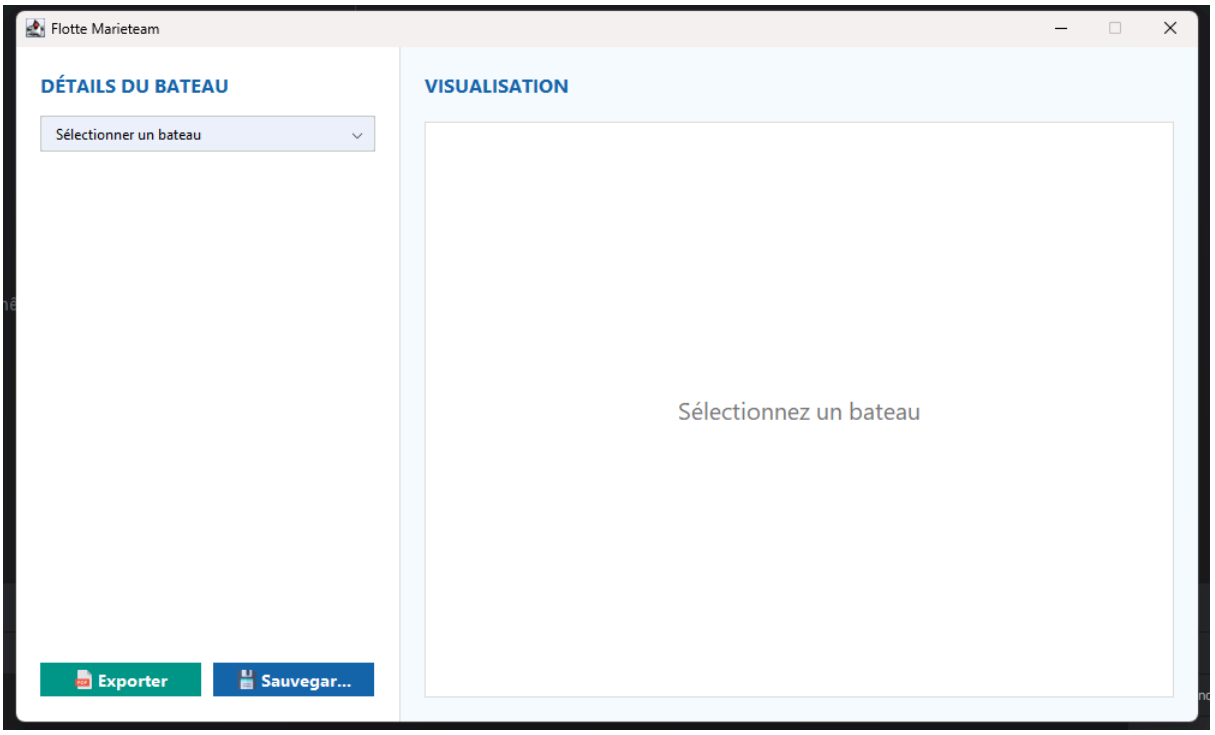
Pour instancier une Interface, il faut mettre en arguement une HashMap de bateaux et une List d'équipement. Dans notre cas dans la méthode Main, le

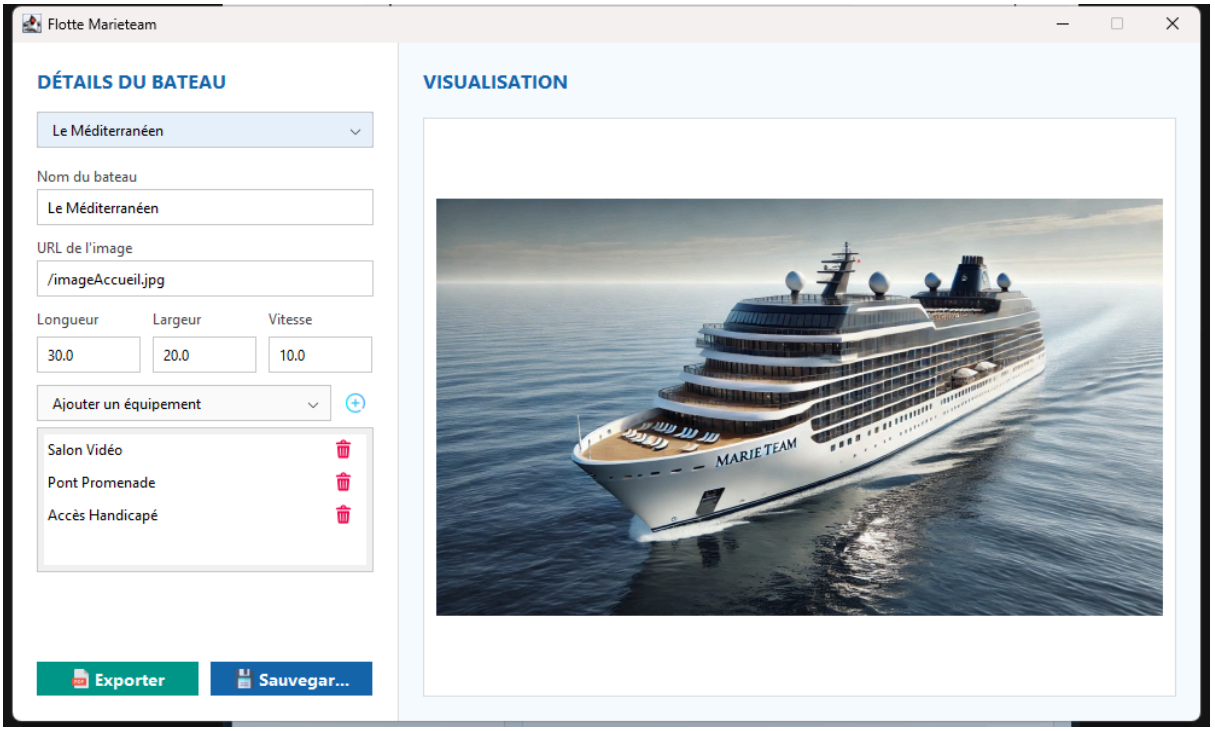
finalBateaux est la totalité de la flotte en base de données et la liste equipementDispo est la liste de tous les équipements disponibles en base de données.

Ils sont instanciés ainsi:

```
HashMap<Integer, BateauVoyageur> bateaux = new HashMap<>();
        bateaux = databaseQuery.SelectAllBateau();
        HashMap<Integer, BateauVoyageur> finalBateaux = bateaux;
        List<Equipement> equipementDispo = DatabaseQuery.SelectAllEquipe
ment();
```

(Pour plus de précision sur databaseQuery, voir la partie sur la base de données)





IV. Base de données

La base de données est la même que le projet web, celle sur Supabase. Pour y accèder et faire des requêtes dessus, il a fallu installer la dépendance postgresql pour Java.

On effectue sa connexion dans la classe DatabaseConnection de la manière suivant:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class DatabaseConnection {
    // Copie ta string JDBC ici (trouvée dans Supabase > Settings > Databas
e)
    private static final String JDBC_URL = "url trouvable sur supabase";


    private static Connection connection;

    public static Connection getConnection() throws SQLException {
        if (connection == null || connection.isClosed()) {
```

```
            connection = DriverManager.getConnection(JDBC_URL);
        }
        return connection;
    }
}
```

Ensuite dans la classe DatabaseQuery, on peut y déclarer toutes les requêtes nécessaires à l'appli Java, voici par exemple la requête pour avoir tous les bateaux:

```
public static HashMap<Integer,BateauVoyageur> SelectAllBateau() {
        HashMap<Integer,BateauVoyageur> bateaux = new HashMap<>();
        try {
            Connection conn = DatabaseConnection.getConnection();
            System.out.println("✅ Connexion réussie à Supabase !");
            Statement statement = conn.createStatement();
            String query = "SELECT * FROM bateau";
            ResultSet resultSet = statement.executeQuery(query);

            try {
                while (resultSet != null && resultSet.next()) {
                    List<Equipement> equipements = SelectBateauEquipment(resu
ltSet.getInt("id") );
                    BateauVoyageur bateauVoyageur = new BateauVoyageur(
                            resultSet.getInt("id"),
                            resultSet.getString("nom"),
                            resultSet.getDouble("largeur"),
                            resultSet.getDouble("longueur"),
                            resultSet.getInt("vitesse"),
                            equipements,
                            resultSet.getString("ImageUrl")
                    );
                    bateaux.put(resultSet.getInt("id"),bateauVoyageur);
                }
            }
            catch (SQLException e) {
                System.out.println("❌ Erreur de connexion ou d'exécution de la r
```

```
equête : " + e.getMessage());
            e.printStackTrace();
        }
        conn.close(); // facultatif, sauf si tu veux réutiliser plus tard
    } catch (
            SQLException e) {
            System.out.println("❌ Erreur de connexion : " + e.getMessage());
             e.printStackTrace();
    }
    return bateaux;
  }
```

Dans la classe Main, il faut bien déclarer une instance de DatabaseQuery pour permettre à l'appli de faire des requêtes:

```
DatabaseQuery databaseQuery = new DatabaseQuery();
```

Dans la méthode main par exemple on s'en sert comme suit:

```
HashMap<Integer, BateauVoyageur> bateaux = new HashMap<>();
      bateaux = databaseQuery.SelectAllBateau();
      HashMap<Integer, BateauVoyageur> finalBateaux = bateaux;
      List<Equipement> equipementDispo = DatabaseQuery.SelectAllEquipe
ment();
```

V. Création de PDF

La création d'un pdf contenant toute la flotte Marieteam est gérée par la classe Pdf. Elle est appelée dans la classe Interface, lors de l'évenement click sur le bouton créer un pdf.

```
buttonPdf.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            String userHome = System.getProperty("user.home");
            String filePath = userHome + "/Desktop/FlotteMarieteam.pdf";
            try {
                Pdf.createPdf(filePath, bateaux);
```

```
                int option = JOptionPane.showOptionDialog(Interface.this,
                        "<html><div style='font-family:Segoe UI;font-size:12pt;pa
dding:10px'>" +
                                "PDF créé avec succès!<br>Souhaitez-vous ouvrir le
document?</div></html>",
                        "Export réussi",
                        JOptionPane.YES_NO_OPTION,
                        JOptionPane.QUESTION_MESSAGE,
                        null,
                        new Object[]{"Ouvrir", "Fermer"},
                        "Ouvrir");

                if (option == JOptionPane.YES_OPTION) {
                    if (Desktop.isDesktopSupported()) {
                        Desktop.getDesktop().open(new File(filePath));
                    }
                }

            } catch (Exception exception) {
                JOptionPane.showMessageDialog(Interface.this,
                        "<html><div style='font-family:Segoe UI;font-size:12pt;pa
dding:10px'>" +
                                "Erreur lors de la création du PDF:<br>" + exception.
getMessage() + "</div></html>",
                        "Erreur",
                        JOptionPane.ERROR_MESSAGE);
                exception.printStackTrace();
            }
        }
    });
```

La méthode prend en argument un chemin pour enregistrer le fichier et une HashMap contenant les bateaux. Ici le fichier est automatiquement enregistré sr le bureau avec:

```
String userHome = System.getProperty("user.home");
            String filePath = userHome + "/Desktop/FlotteMarieteam.pdf";
```

Ici on prend en argument la HashMap de bateaux que l'on a récupérée lors de l'instanciation de notre interface. A noter que si elle est mise à jour par l'utilisateur, alors celle prise en compte sera elle aussi à jour pour la création du pdf.

VI. Mettre à jour un bateau

Un utilisateur peut modifier les informations d'un bateau et les enregistrer en base de données. Cette logique est contenu dans Interface et fait appel à la requête update contenu dans DatabaseQuery.

Voici la méthode pour prendre en compte les nouvelles informations, vérifiées au préalable:

```
buttonSave.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            DatabaseQuery databaseQuery = new DatabaseQuery();
            BateauVoyageur b = (BateauVoyageur) comboBox.getSelectedIte
m();
            if (b == null) return;

            String nomBateau = nameField.getText().trim();

            try {
                Double longueurBateau = Double.parseDouble(longueurField.g
etText().trim());
                Double largeurBateau = Double.parseDouble(largeurField.getTe
xt().trim());
                Double vitesseBatVoy = Double.parseDouble(vitesseField.getT
ext().trim());
                String image = urlField.getText().trim();

                if (nomBateau.isEmpty() || longueurBateau < 0 || largeurBateau
< 0 || vitesseBatVoy < 0) {
                    JOptionPane.showMessageDialog(Interface.this,
```

```
                    "<html><div style='font-family:Segoe UI;font-size:12pt;
padding:10px'>" +
                            "Veuillez remplir tous les champs.</div></html>",
                    "Erreur",
                    JOptionPane.ERROR_MESSAGE);
                return;
            }

            // Mise à jour du bateau
            Boolean update = databaseQuery.updateBateau(b.getIdBateau
(), nomBateau, largeurBateau, longueurBateau,
                    vitesseBatVoy, image);

            if (update) {
                // Mise à jour complète de l'objet
                b.setNomBateau(nomBateau);
                b.setLongueurBateau(longueurBateau);
                b.setLargeurBateau(largeurBateau);
                b.setVitesseBatVoy(vitesseBatVoy);
                b.setImage(image);

                // Mise à jour des équipements
                // D'abord supprimer tous les équipements existants pour ce
bateau
                databaseQuery.deleteAllEquipements(b.getIdBateau());

                // Puis ajouter les nouveaux équipements
                boolean allEquipmentsSaved = true;
                for (Equipement equip : currentEquipements) {
                    boolean saved = databaseQuery.addEquipementToBateau(
                            b.getIdBateau(),
                            equip.getIdEquip()
                    );
                    if (!saved) {
                        allEquipmentsSaved = false;
                    }
                }
```

```
                // Mettre à jour la liste des équipements dans l'objet bateau
                b.setEquipements(new ArrayList<>(currentEquipements));

                // Mettre à jour l'objet dans la HashMap
                bateaux.put(b.getIdBateau(), b);

                // Rafraîchir correctement la comboBox
                int selectedIndex = comboBox.getSelectedIndex();
                comboBox.removeItemAt(selectedIndex);
                comboBox.insertItemAt(b, selectedIndex);
                comboBox.setSelectedIndex(selectedIndex);

                String message = "<html><div style='font-family:Segoe UI;f
ont-size:12pt;padding:10px'>" +
                        "Les modifications ont été sauvegardées.";

                if (!allEquipmentsSaved) {
                    message += "<br><br><span style='color:orange'>Attenti
on: Certains équipements n'ont pas pu être sauvegardés.</span>";
                }

                message += "</div></html>";

                JOptionPane.showMessageDialog(Interface.this,
                        message,
                        "Sauvegarde",
                        JOptionPane.INFORMATION_MESSAGE);
            } else {
                JOptionPane.showMessageDialog(Interface.this,
                        "<html><div style='font-family:Segoe UI;font-size:12pt;
padding:10px'>" +
                                "Une erreur est survenue lors de la sauvegarde des
modifications.</div></html>",
                        "Erreur",
                        JOptionPane.ERROR_MESSAGE);
            }
        } catch (NumberFormatException ex) {
            JOptionPane.showMessageDialog(Interface.this,
```

```
                    "<html><div style='font-family:Segoe UI;font-size:12pt;pa
dding:10px'>" +
                            "Veuillez entrer des valeurs numériques valides pour l
a longueur, largeur et vitesse.</div></html>",
                    "Erreur",
                    JOptionPane.ERROR_MESSAGE);
            }
        }
    });
```

L'appel à la base de données se fait ici :

```
 Boolean update = databaseQuery.updateBateau(b.getIdBateau(), nomBate
au, largeurBateau, longueurBateau,
                    vitesseBatVoy, image);
```

Et dans la classe DatabaseQuery est utilisé de ce fait cette méthode:

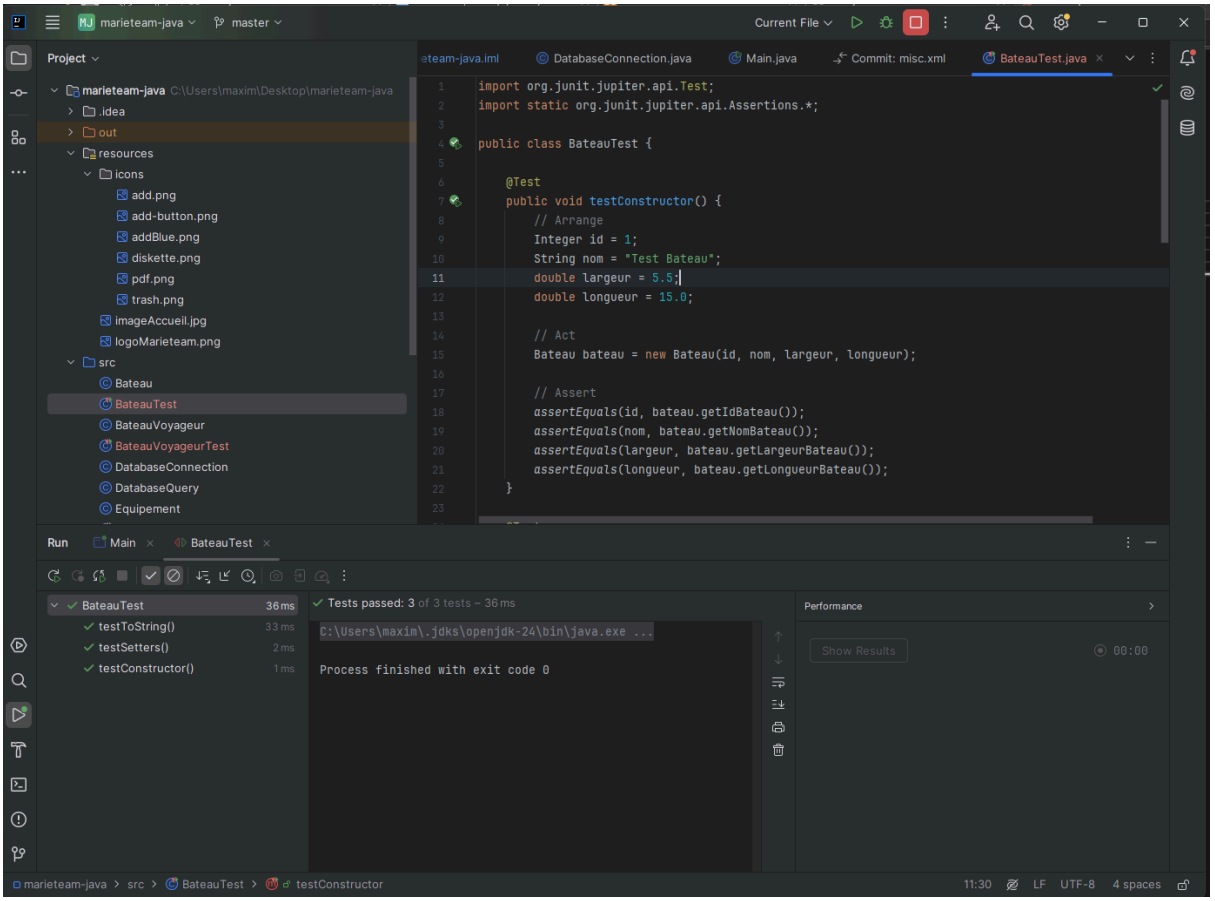
```
 public static Boolean updateBateau(int id, String nom, double largeur, doub
le longueur, double vitesse, String image) {

    try {
        Connection conn = DatabaseConnection.getConnection();
        System.out.println("✅ Connexion réussie à Supabase !");
        System.out.println(id);
        String query = "UPDATE bateau SET nom = '" + nom + "', largeur = "
+ largeur +
                ", longueur = " + longueur + ", vitesse = " + vitesse + ", ImageU
rl = '" + image + "'" +
                " WHERE id = " + id;

        try {
            // Utilisation de Statement à la place de PreparedStatement
            Statement statement = conn.createStatement();
            int rowsAffected = statement.executeUpdate(query);
            conn.close(); // Facultatif sauf si vous voulez réutiliser plus tard
```

```
            return rowsAffected > 0;
        } catch (SQLException e) {
            System.err.println("❌ Erreur lors de la mise à jour : " + e.getMess
age());
            e.printStackTrace();
            return false; // Si une erreur survient
        }




    } catch (
            SQLException e) {
        System.out.println("❌ Erreur de connexion : " + e.getMessage());
        e.printStackTrace();
        return false;
    }


}
```

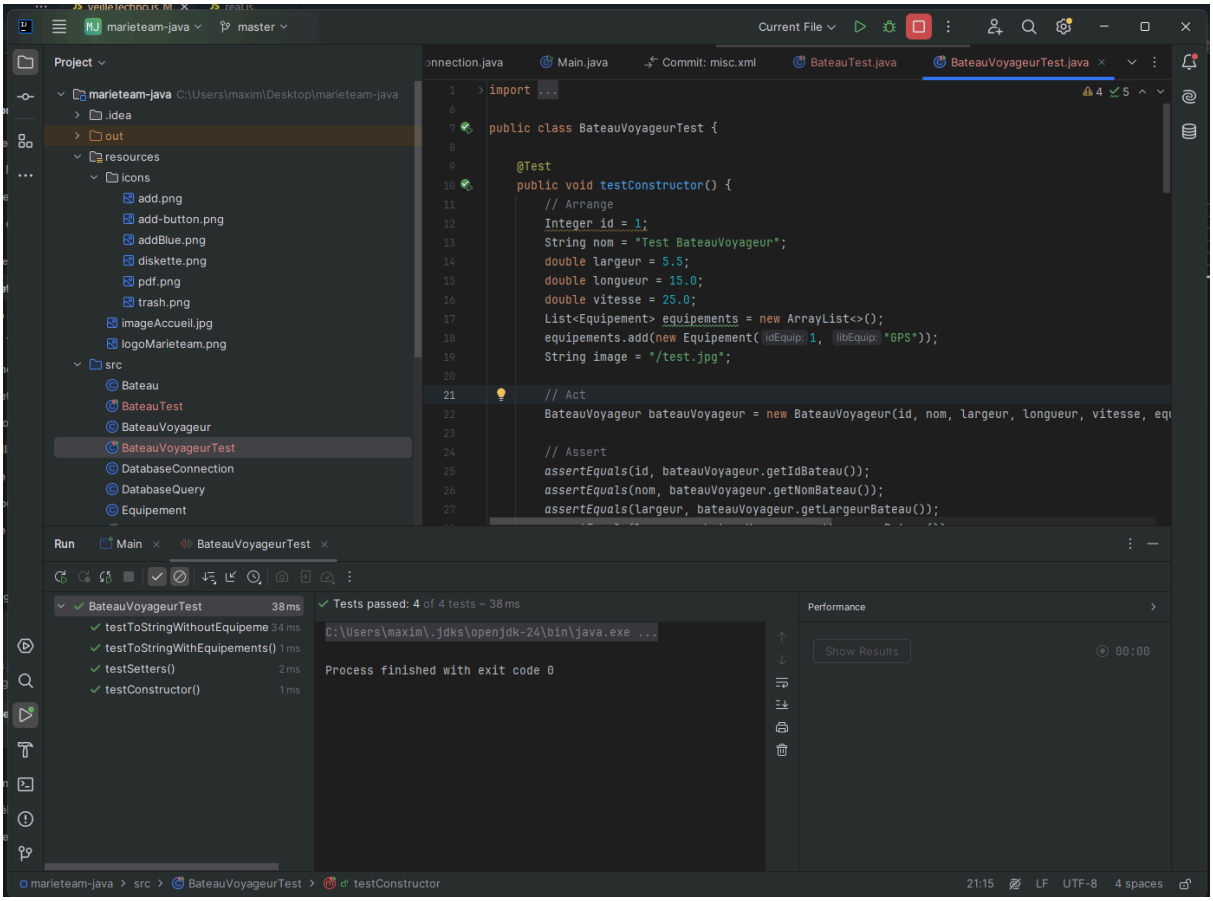
VI. Test Unitaires

Voici l'ensemble des tests unitaires réalisés avec Junit:

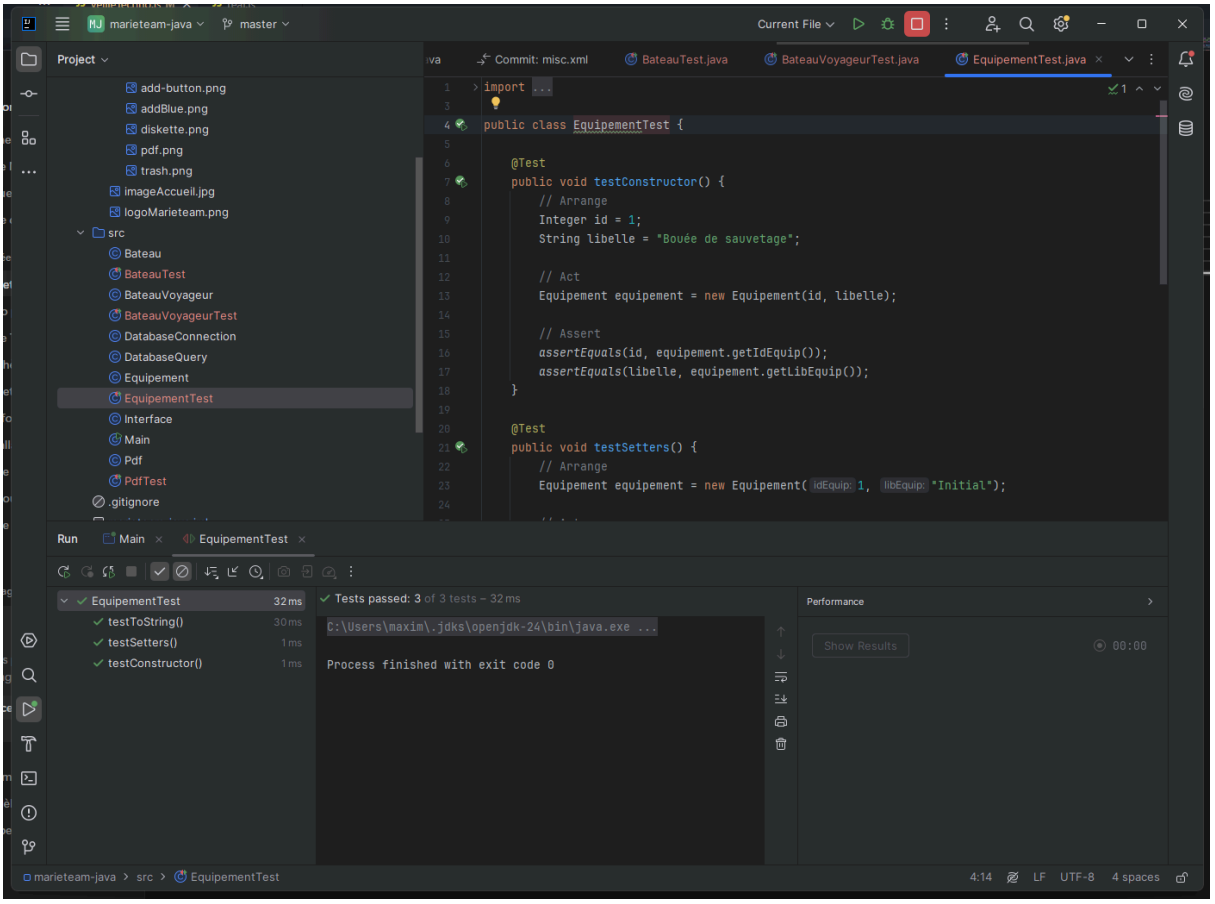
BateauTest:



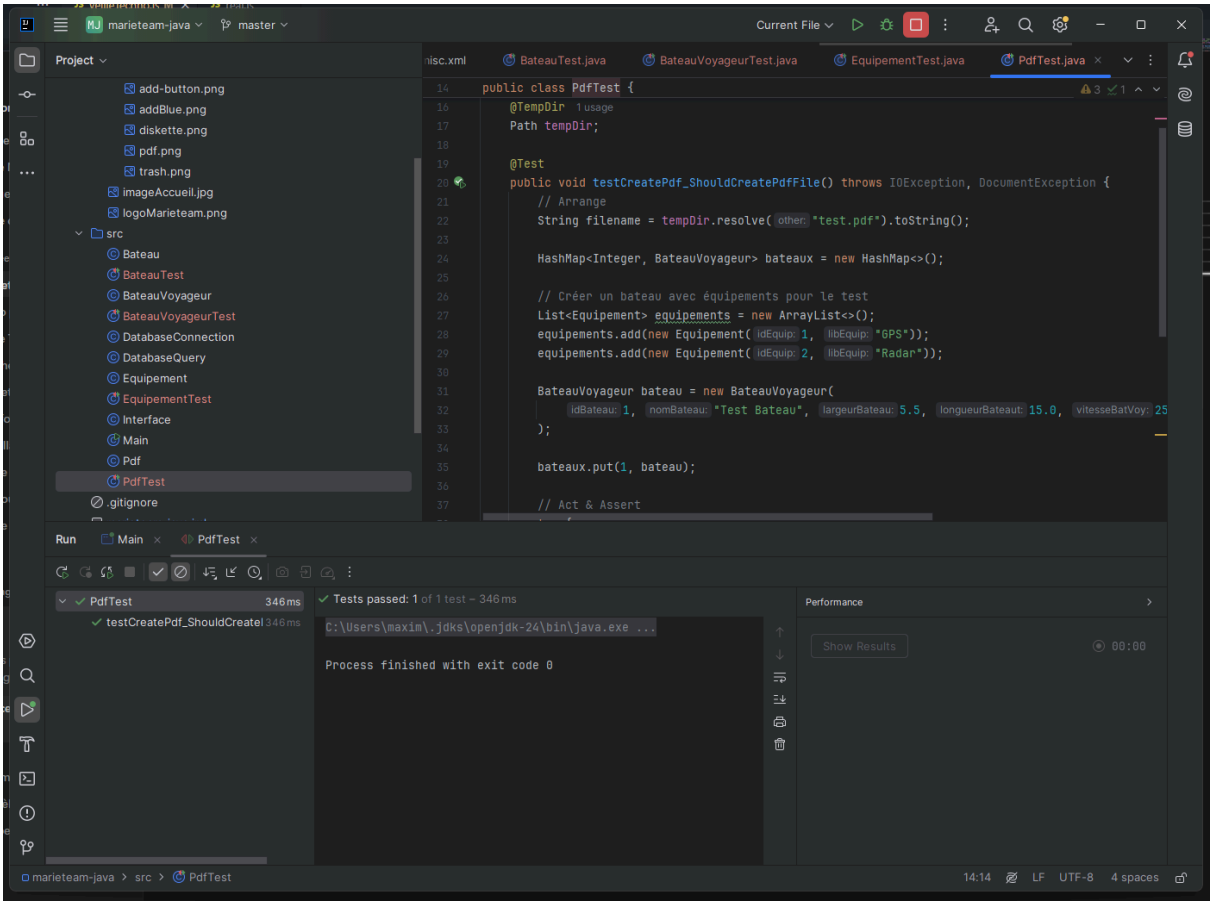
BateauVoyageurTest:



EquipementTest:

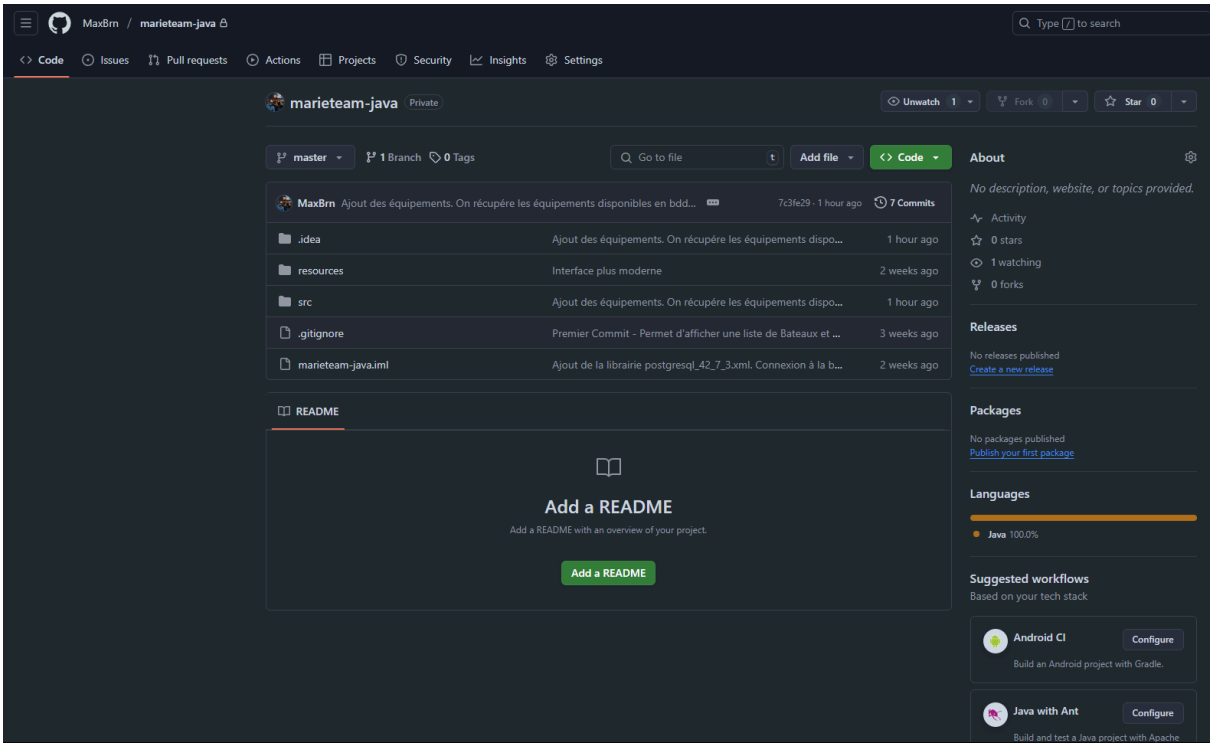

marieteam-java
master
Current File
Project
add-button.png
addBlue.png
diskette.png
pdf.png
trash.png
imageAccueil.jpg
logoMarieteam.png
src
Bateau
BateauTest
BateauVoyageur
BateauVoyageurTest
DatabaseConnection
DatabaseQuery
Equipement
EquipementTest
Interface
Main
Pdf
PdfTest
.gitignore
Commit: misc.xml
BateauTest.java
BateauVoyageurTest.java
EquipementTest.java
import ...
public class EquipementTest {
@Test
public void testConstructor() {
// Arrange
Integer id = 1;
String libelle = "Bouée de sauvetage";
// Act
Equipement equipement = new Equipement(id, libelle);
// Assert
assertEquals(id, equipement.getIdEquip());
assertEquals(libelle, equipement.getLibEquip());
}
@Test
public void testSetters() {
// Arrange
Equipement equipement = new Equipement(idEquip: 1, libEquip: "Initial");
Run
Main
EquipementTest
EquipementTest 32 ms
testToString() 30 ms
testSetters() 1 ms
testConstructor() 1 ms
Tests passed: 3 of 3 tests – 32 ms
C:\Users\maxim\.jdks\openjdk-24\bin\java.exe ...
Process finished with exit code 0
Performance
Show Results
00:00
marieteam-java
src
EquipementTest
4:14
LF
UTF-8
4 spaces

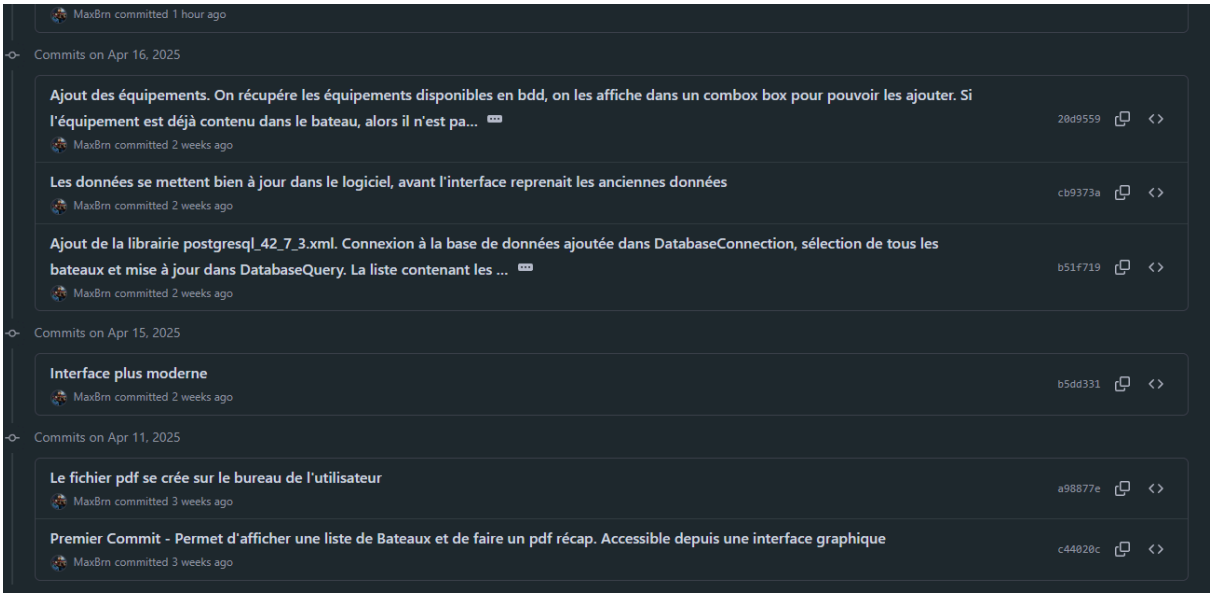
PdfTest:



VII. Gestion du versionning

Le projet est enregistré sur un dépôt distant Git afin de mieux gérer le versionning et la collaboration entre les membres du groupe.


MaxBrn / marieteam-java
Type / to search
Code
Issues
Pull requests
Actions
Projects
Security
Insights
Settings
marieteam-java
Private
Unwatch 1
Fork 0
Star 0
master
1 Branch
0 Tags
Go to file
t
Add file
Code
About
No description, website, or topics provided.
MaxBrn Ajout des équipements. On récupére les équipements disponibles en bdd... 7c3fe29 · 1 hour ago 7 Commits
.idea Ajout des équipements. On récupére les équipements dispo... 1 hour ago
resources Interface plus moderne 2 weeks ago
src Ajout des équipements. On récupére les équipements dispo... 1 hour ago
.gitignore Premier Commit - Permet d'afficher une liste de Bateaux et ... 3 weeks ago
marieteam-java.iml Ajout de la librairie postgresql_42_7_3.xml. Connexion à la b... 2 weeks ago
Activity
0 stars
1 watching
0 forks
Releases
No releases published
Create a new release
README
Add a README
Add a README with an overview of your project.
Add a README
Packages
No packages published
Publish your first package
Languages
Java 100.0%
Suggested workflows
Based on your tech stack
Android CI
Configure
Build an Android project with Gradle.
Java with Ant
Configure


MaxBrn committed 1 hour ago
Commits on Apr 16, 2025
Ajout des équipements. On récupére les équipements disponibles en bdd, on les affiche dans un combox box pour pouvoir les ajouter. Si l'équipement est déjà contenu dans le bateau, alors il n'est pa...
MaxBrn committed 2 weeks ago
20d9559
Les données se mettent bien à jour dans le logiciel, avant l'interface reprenait les anciennes données
MaxBrn committed 2 weeks ago
cb9373a
Ajout de la librairie postgresql_42_7_3.xml. Connexion à la base de données ajoutée dans DatabaseConnection, sélection de tous les bateaux et mise à jour dans DatabaseQuery. La liste contenant les ...
MaxBrn committed 2 weeks ago
b51f719
Commits on Apr 15, 2025
Interface plus moderne
MaxBrn committed 2 weeks ago
b5dd331
Commits on Apr 11, 2025
Le fichier pdf se crée sur le bureau de l'utilisateur
MaxBrn committed 3 weeks ago
a98877e
Premier Commit - Permet d'afficher une liste de Bateaux et de faire un pdf récap. Accessible depuis une interface graphique
MaxBrn committed 3 weeks ago
c44020c