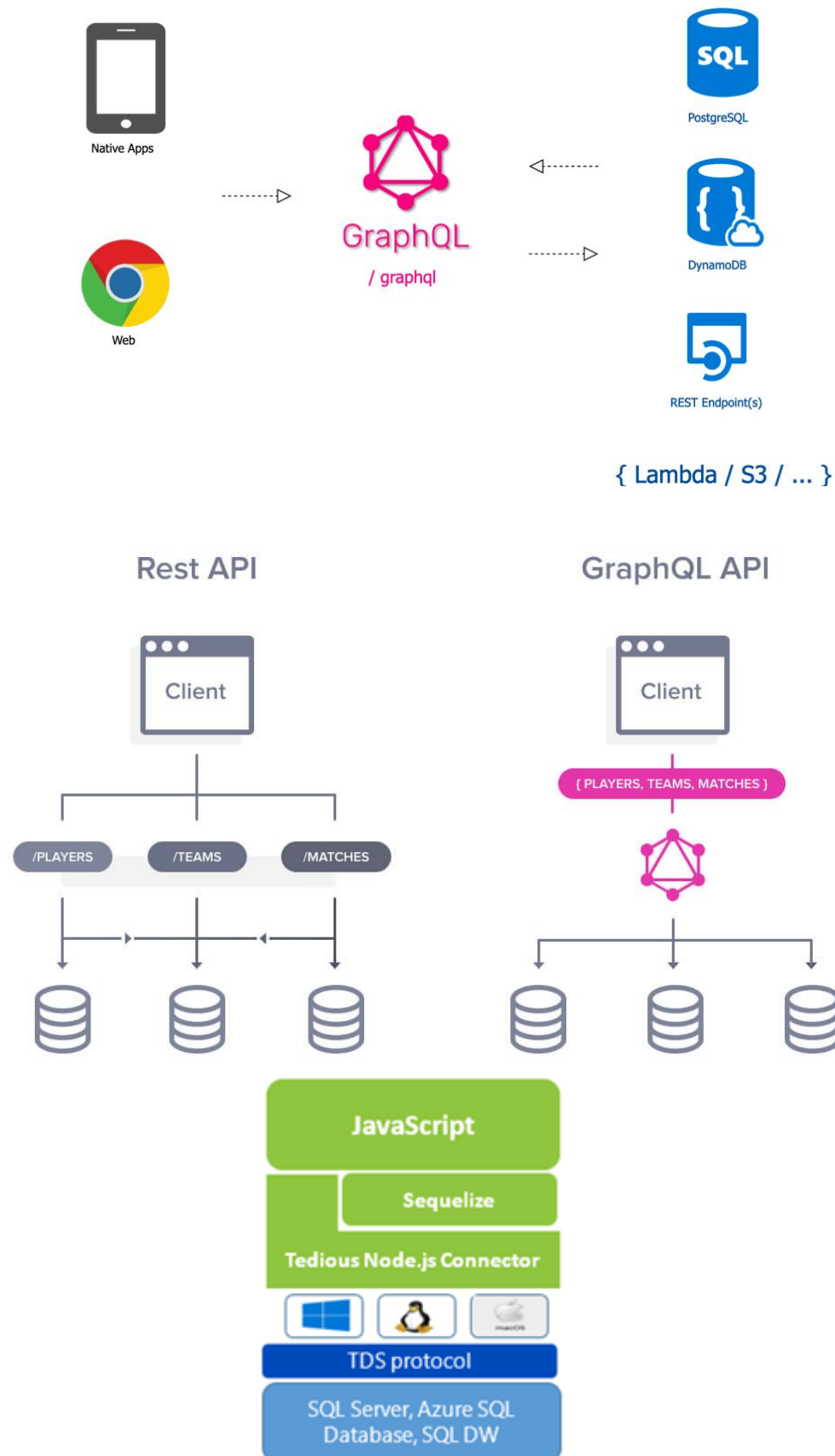


API built with NodeJs and express, GraphQL and SQLServer



Folder structure:

```

  graphql-sqlserver
  > build
  > node_modules
  > src
    > config
      {} config.json
    > dbhelper
      JS saveList.js
    > express
    > graphql
      > input
        JS input_list.js
      > mutations
        JS mutations.js
      > output
        JS output_list.js
        JS token.js
      > rootquery
        JS RootQuery.js
      JS schema.js
    > models
      JS index.js
      JS list.js
  6 .babelrc
  .gitignore
  JS index.js
  {} package.json
  README.md
  webpack.config.js
  varn.lock
```

Modules requires:

```
"dependencies": {
  "babel-cli": "^6.26.0",
  "babel-preset-env": "^1.7.0",
  "babel-preset-stage-3": "^6.24.1",
  "bcrypt": "^3.0.6",
  "body-parser": "^1.19.0",
  "cross-env": "^5.2.1",
  "express": "^4.17.1",
  "express-graphql": "^0.9.0",
  "express-session": "^1.16.2",
  "graphql": "^14.5.5",
  "jsonwebtoken": "^8.5.1",
  "nodemon": "^1.19.2",
  "rimraf": "^3.0.0",
  "sequelize": "^5.18.4",
  "sequelize-cli": "^5.5.1",
  "tedious": "^6.4.0",
  "webpack": "^4.40.2",
  "webpack-cli": "^3.3.8"
```

Project Setup

Step 1: Create node project

npm init -y

Step 2: Add dependencies

yarn add body-parser express tedious sequelize sequelize-cli graphql bcrypt express-graphql nodemon jsonwebtoken babel-cli babel-preset-env babel-preset-stage-3 webpack webpack-cli

Step 3: create sequelize skeleton configuration

yarn sequelize-skeleton
update config.json file

Step 4: ORM models

Let's create our ORM data models

Step 5: GraphQL Schema

Let's create our GraphQL Schema

Step 6: express server

Let's create our server

Step 7: run

"start": "nodemon --exec babel-node index.js --offline",
Yarn start

In Detailed: configuration.js

```
config.json ×
graphql-sqlserver > src > config > config.json > {} development
1  {
2    "development": {
3      "dialect": "mssql",
4      "username": "sa",
5      "password": "java",
6      "database": "test",
7      "host": "DWJHBDBA010",
8      "dialectOptions": {
9        "options": {
10         "instanceName": "MSSQLSERVER16",
11         "useUTC": false,
12         "dateFirst": 1
13       }
14     },
15     "define": {
16       "timestamps": false,
17       "freezeTableName": true
18     }
19   },
```

Model: configuration

```
JS index.js ×
graphql-sqlserver > src > models > JS index.js > forEach() callback
1  'use strict';
2
3  var fs      = require('fs');
4  var path    = require('path');
5  const Sequelize = require('sequelize'); //Sequelize is a promise-based Node.js ORM for Micro
6  var basename = path.basename(__filename);
7  var env      = process.env.NODE_ENV || 'development';
8  var config   = require(__dirname + '/../config/config.json')[env];
9  var db       = {};
10
11  if (config.use_env_variable) {
12    var sequelize = new Sequelize(process.env[config.use_env_variable], config);
13  } else {
14    var sequelize = new Sequelize(config.database, config.username, config.password, config);
15  }
16
17  fs
18    .readdirSync(__dirname)
19    .filter(file => {
20      return (file.indexOf('.') !== 0) && (file !== basename) && (file.slice(-3) === '.js');
21    })
22    .forEach(file => {
23      var model = sequelize['import'](path.join(__dirname, file));
24      db[model.name] = model;
25    });
26
27  Object.keys(db).forEach(modelName => {
28    if (db[modelName].associate) {
29      db[modelName].associate(db);
30    }
31  });
32
33  db.sequelize = sequelize;
34  db.Sequelize = Sequelize;
35
36  module.exports = db;
```

List model:

```
JS list.js ×
graphql-sqlserver > src > models > JS list.js > <function>
1  export default (sequelize, DataTypes) => {
2    const List = sequelize.define('list', {
3      id: {
4        type: DataTypes.INTEGER,
5        primaryKey: true,
6        autoIncrement: true
7      },
8      name: {
9        type: DataTypes.STRING,
10       allowNull: false
11     }
12   },
13   {
14     freezeTableName: true,
15   }
16 );
17 return List;
18 }
```

