**A picture containing graphical user interface

Description automatically generated**

3/28/2021

Interim report

tradegame.org

Current stage of a project: https://github.com/c4rt0/tradeGame/tree/1a6b98051fa824c1eea0796f8093e89aed242c2d

Adam Piasecki

Student number: 20086512

Table of Contents

[**1.** **INTRODUCTION** 2](#_Toc67863214)

[**2.** **FUNCTIONAL DESCRIPTION** 2](#_Toc67863215)

[**3.** **TECHNOLOGY** 3](#_Toc67863216)

[**4.** **DATA MODEL** 8](#_Toc67863217)

[**5.** **ARCHITECTURE** 10](#_Toc67863218)

1. **INTRODUCTION**

Not many ideas seem complex in the initial stage of implementation. It is easy to follow tutorials presented by professionals with many years of significant experience. Steps are logical, logic itself is previously investigated both accordingly to level of understanding and skill of an intentional observer. Nicely introduced tools used during the lectures are previously selected not only from functionality stand point but also considering support, ease of use and documentation. Beginner software developers thrown into the deep ocean of freely floating knowledge face reality which might turn out to be overwhelming. Creativity in the beginning should be rather treated as a wild horse - and that's what I tend to learn during this project a lot. Indeed, nothing can be done without deep understanding of the technology behind the scenes, but fundamental here is planning and execution of crucial milestones which in the end become the tools that we craft.

Trade game is no different. With a few years of exposure to trading, I wanted to create something that would allow me (and others) to practice knowledge gathered over the years on virtual accounts, without a risk of losing real money.

1. **FUNCTIONAL DESCRIPTION**

Originally in a final proposal I stated that trade game will be an online platform which will allow users to take part in an investment contest. Each user will start with the virtual amount of $100,000 and access to live asset data. Users performance (total portfolio) will be visible in comparison to other investors on the score board. Additional option of portfolio reset to original $100,000 value will be also available for those users who would like to start the game again.

That summarizes simple initial game rules.

1. **TECHNOLOGY**

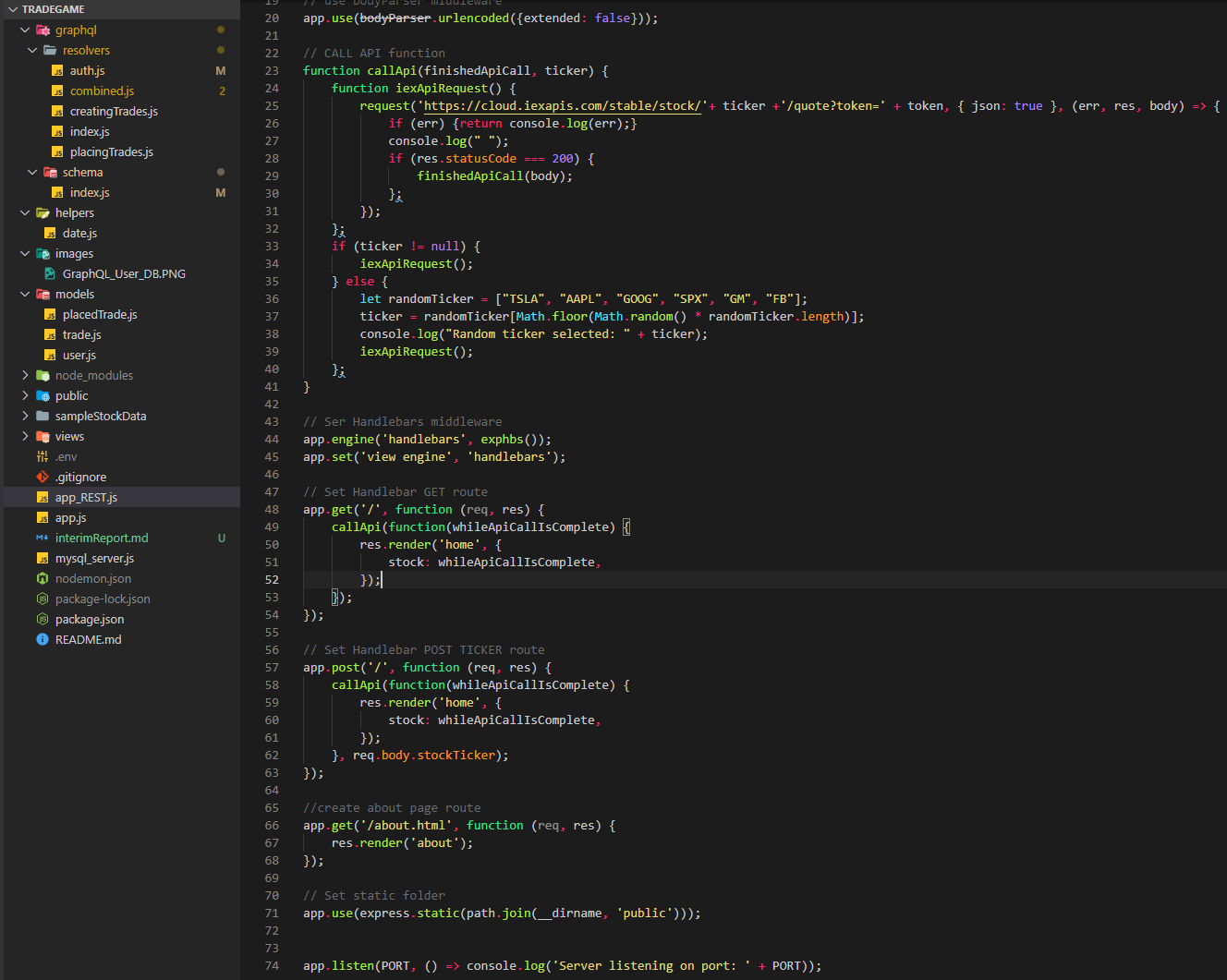
Initial idea behind the application was first to create standard REST API with use of node.js and MongoDB as data base - than implement simple to use REACT client-side interface. That was partially done, I connected to IEX Cloud API with use of **Re**presentational **S**tate **T**ransfer. Thanks to my tutor’s suggestion I stumbled across interesting GraphQL and decided to explore new and unknown yet area to my skillset.  


Figure : Initial implementation of REST API

Off course as the application is not finished, current stage differs from what will it look like in a few weeks – for now however basic GraphQL API with use of Mongoose is implemented and functional. [Link](#LinkToGithub) to current stage of a project is available at the very first page of this report. At it’s simplest, GraphQL is about asking for specific fields on objects   
(source: <https://graphql.org/learn/queries/>). Below presented is a simple schema, with AuthData section I currently work on.

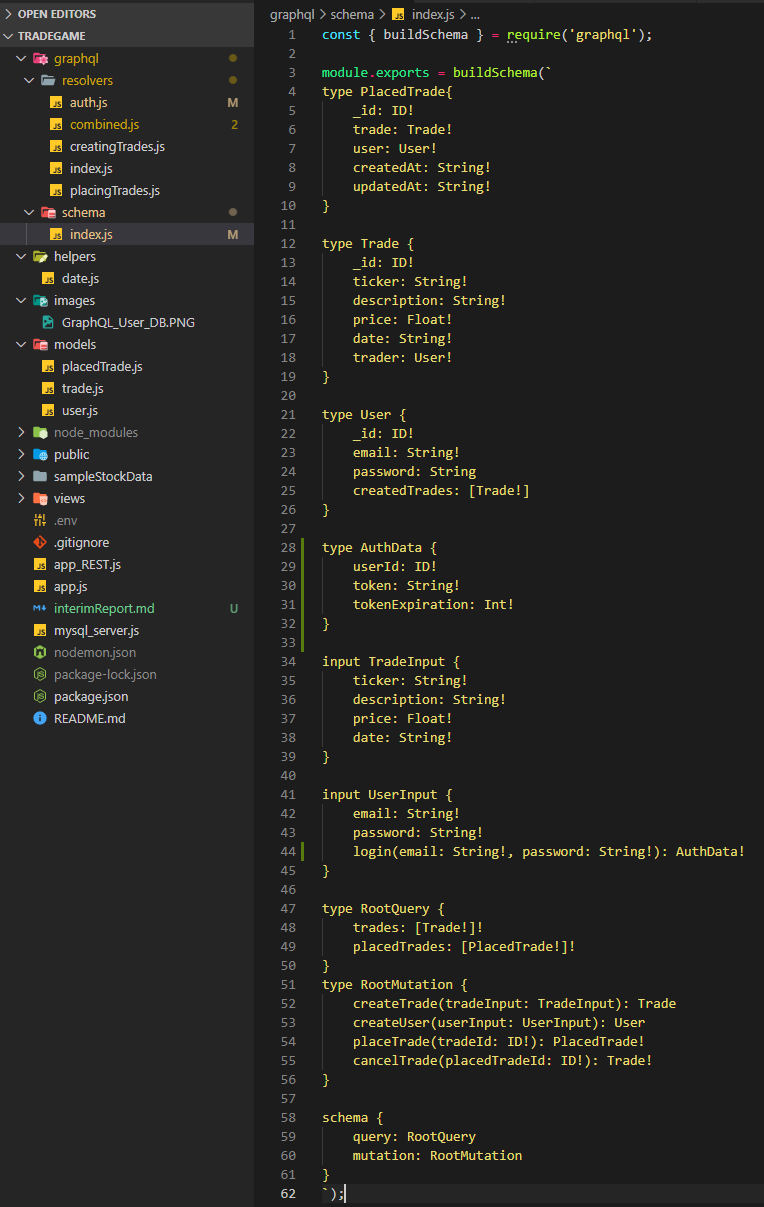


Figure : Current GraphQL Schema

Schema is than passed on to a resolver function and accesses database looking only for arguments specified in a resolver.

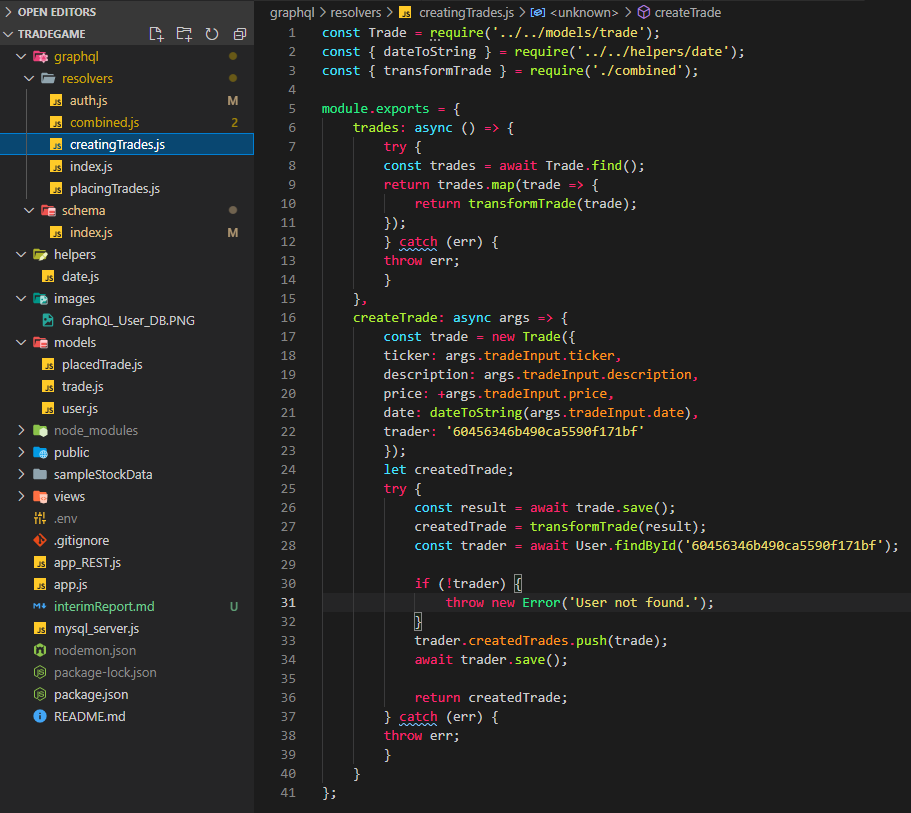


Figure : Current GraphQL "creatingTrade" resolver

Currently my application allows for new trade creation with use of trade.js, placing new trade with placedTrade.js, and new user creation with user.js schema.

In Graphical interface of deployed application two types of queries are allowed :

* query – requesting data

and

* mutation – recording data

Below presented are both requests.

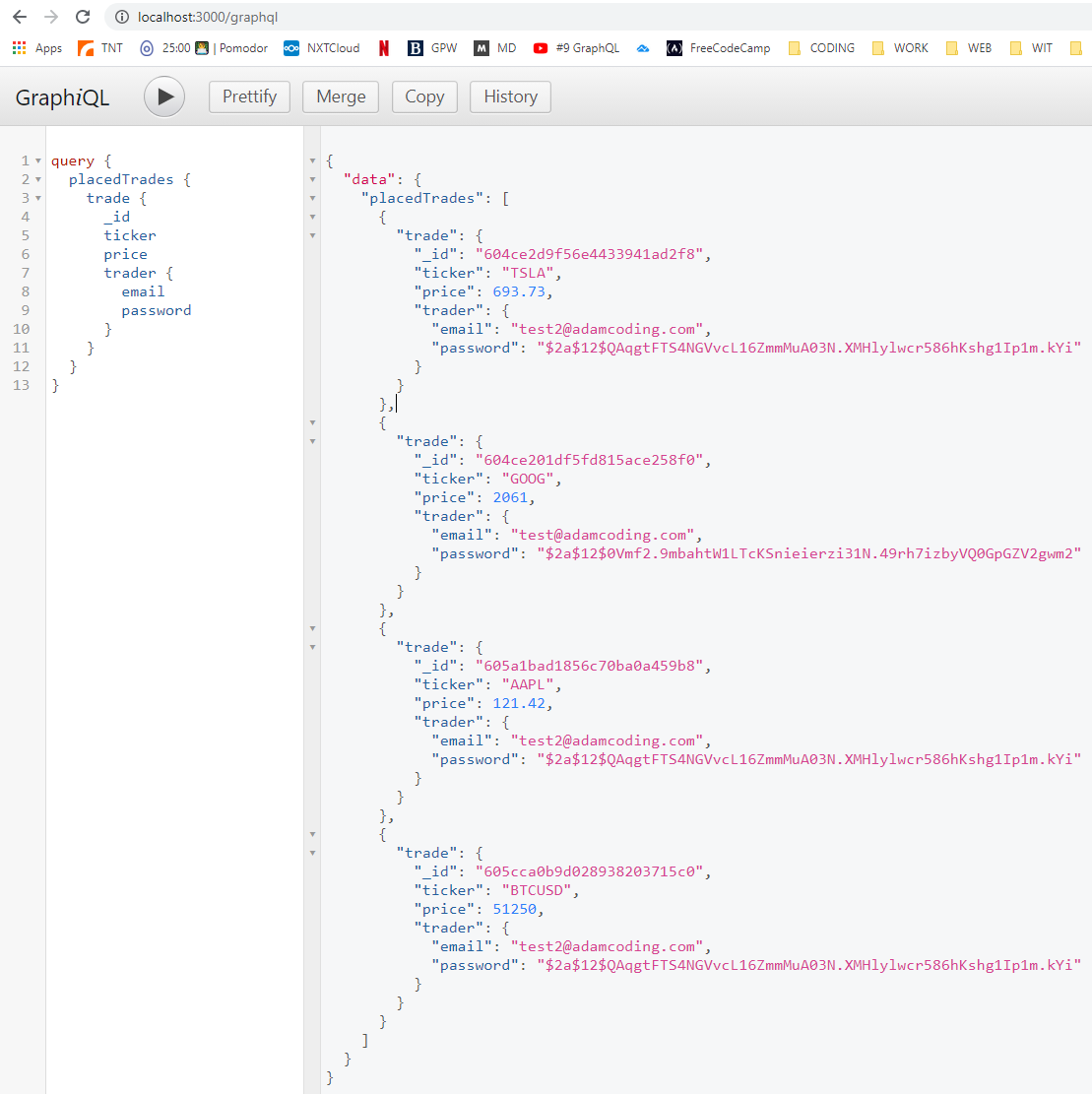


Figure : Data request with use of GraphQL

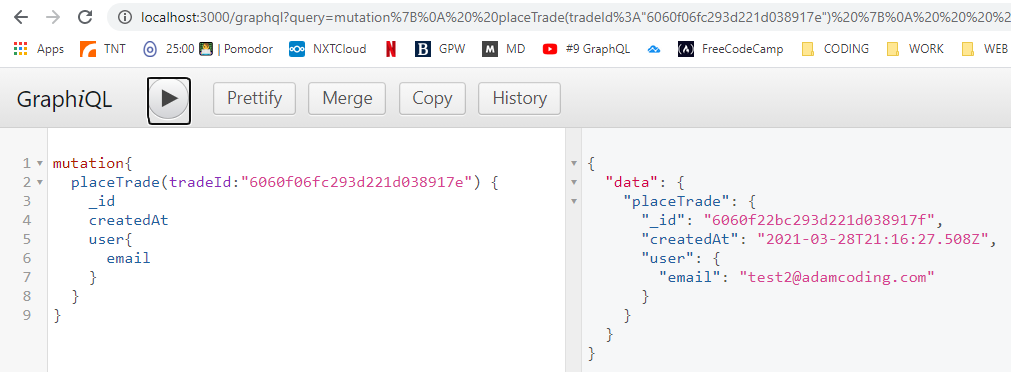


Figure : GraphQL - simple mutation example

1. **DATA MODEL**

Currently as functionality is not yet complete MongoDB cluster presents only three entities:

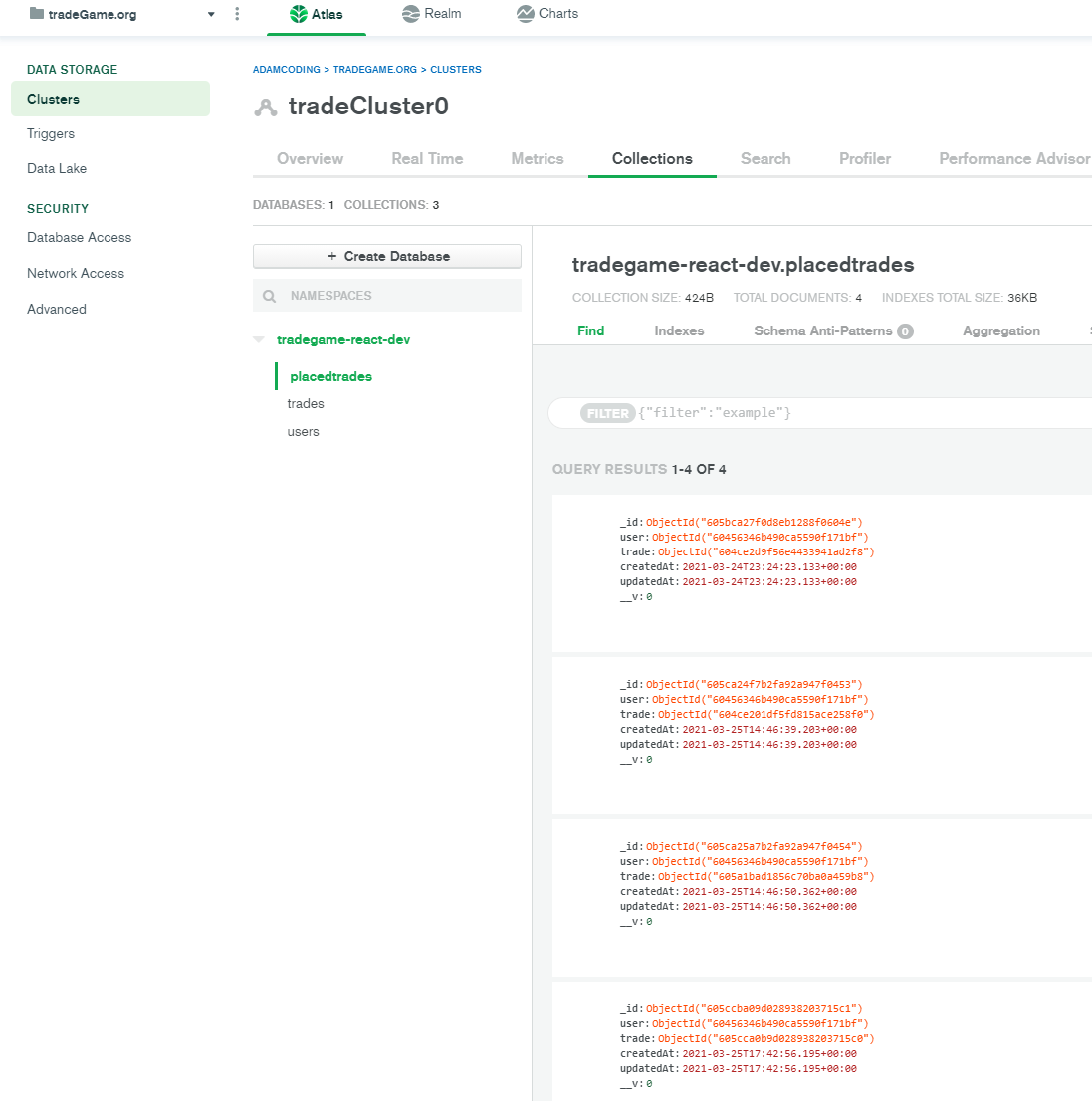


Figure : Current view of MongoDB Cluster

Over time conceptual data model might change according to requirements and deeper understanding of project as a whole, for now however below can be find current vision for final data model.

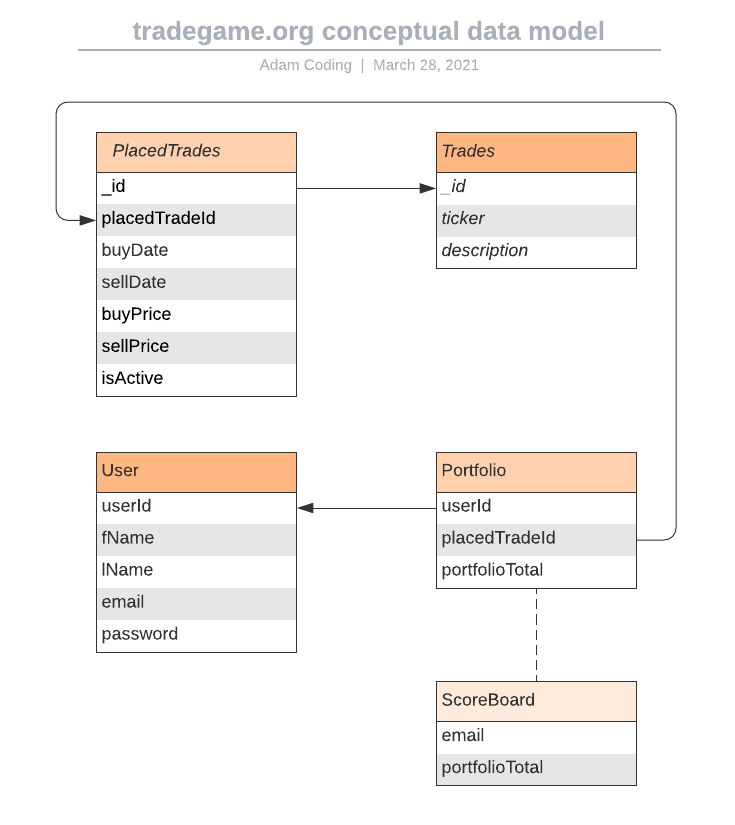


Figure : Conceptual data model

1. **ARCHITECTURE**

As discussed in section of functional description, below diagram presents conceptual architecture of a final product. This interpretation should not differ much from final product.

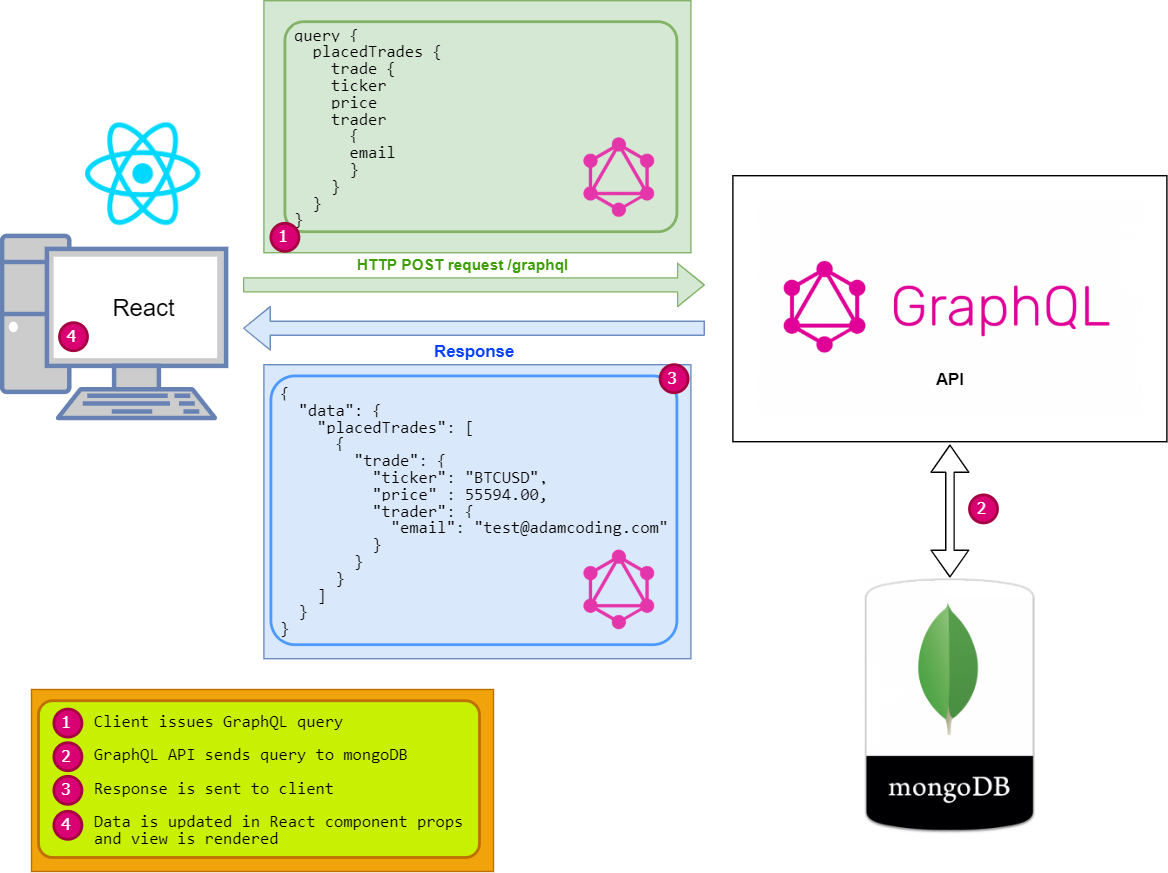


Figure : tradegame.org conceptual architecture