Universitatea Politehnica Timișoara
Facultatea de Automatică și Calculatoare
Departamentul Calculatoare și
Tehnologia Informației

QuizzAppGenerator

Students:
Vladimir NIȚU-ANTONIE
Andrei ROȘU

Master Software Engineering:
2nd Year

Timișoara
2021

Frontend

Installation and configuration

In the case of the front end application, you will need the following programs installed prior on your computer: node and the angular package, this can be done by running npm install angular.

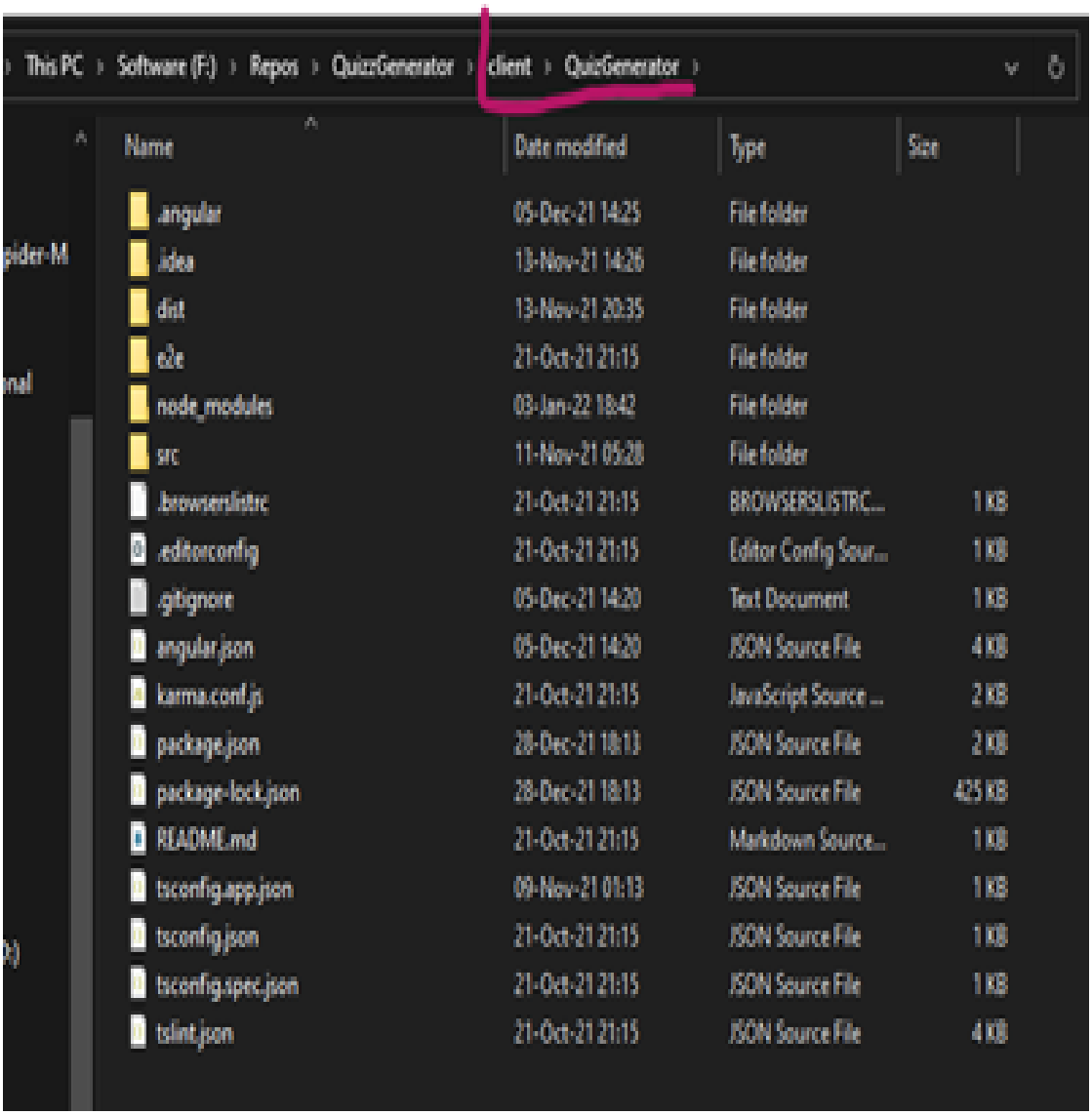

Fig. 1: Installation

After all the prerequired programs are in place go in the main

application folder client/QuizGenerator and run the following command in a terminal ng serve –open. This will build the application and start a webserver at localhost:4200 and will open it for you.

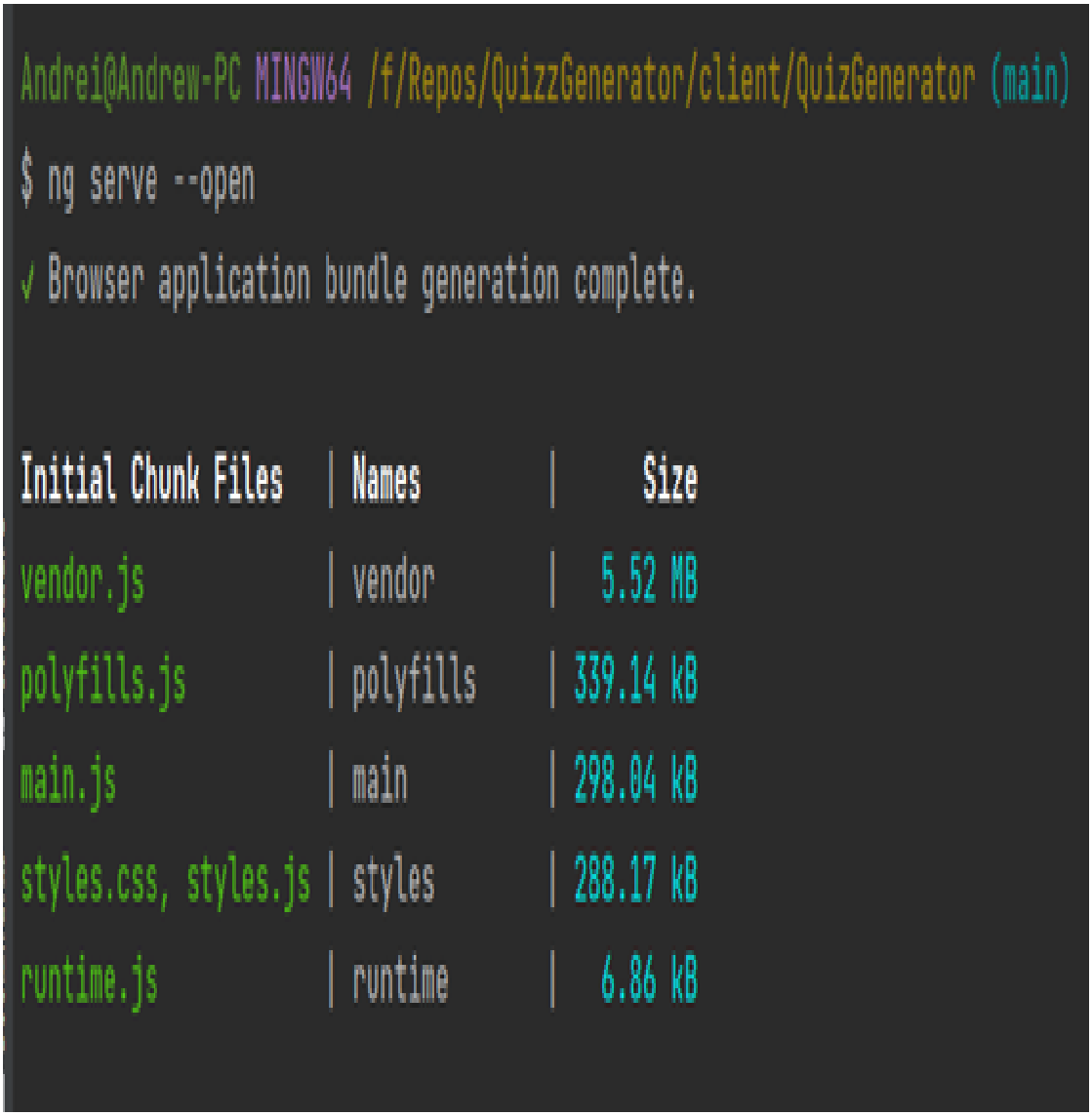

Fig. 2: Terminal

Use cases

The first use case of our application is as follows. The user enters the website and is presented with a welcome page. Next, he has to create a user in order to access all the features of the web application. After logging in dashboard has a few features for the user: easy access to the active questionnaires that he has created, access to the admin created ones, the ability to create a new questionnaire and to see the list of all active ones.

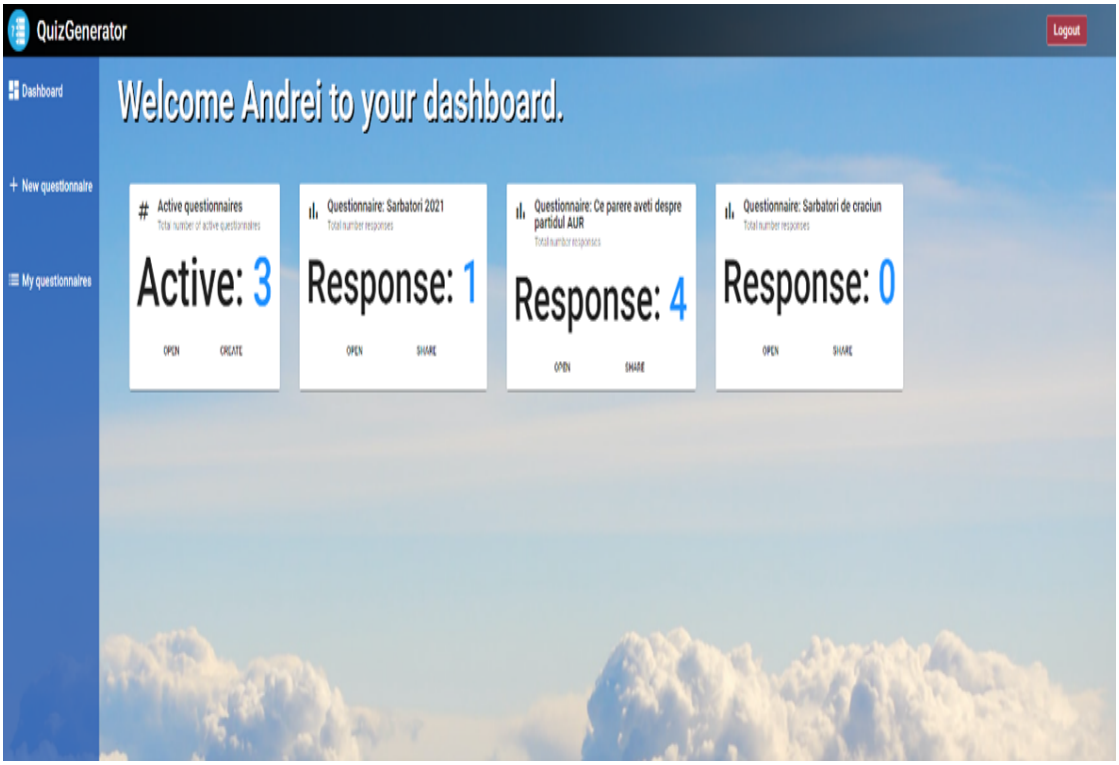

Fig. 3: Dashboard

In the create new questionnaire page the user can create a questionnaire consisting on as many questions and possible answers that he wants when the questionnaire is finished just press the create questionnaire button. It will appear on the bottom of the page when the questionnaire is valid.

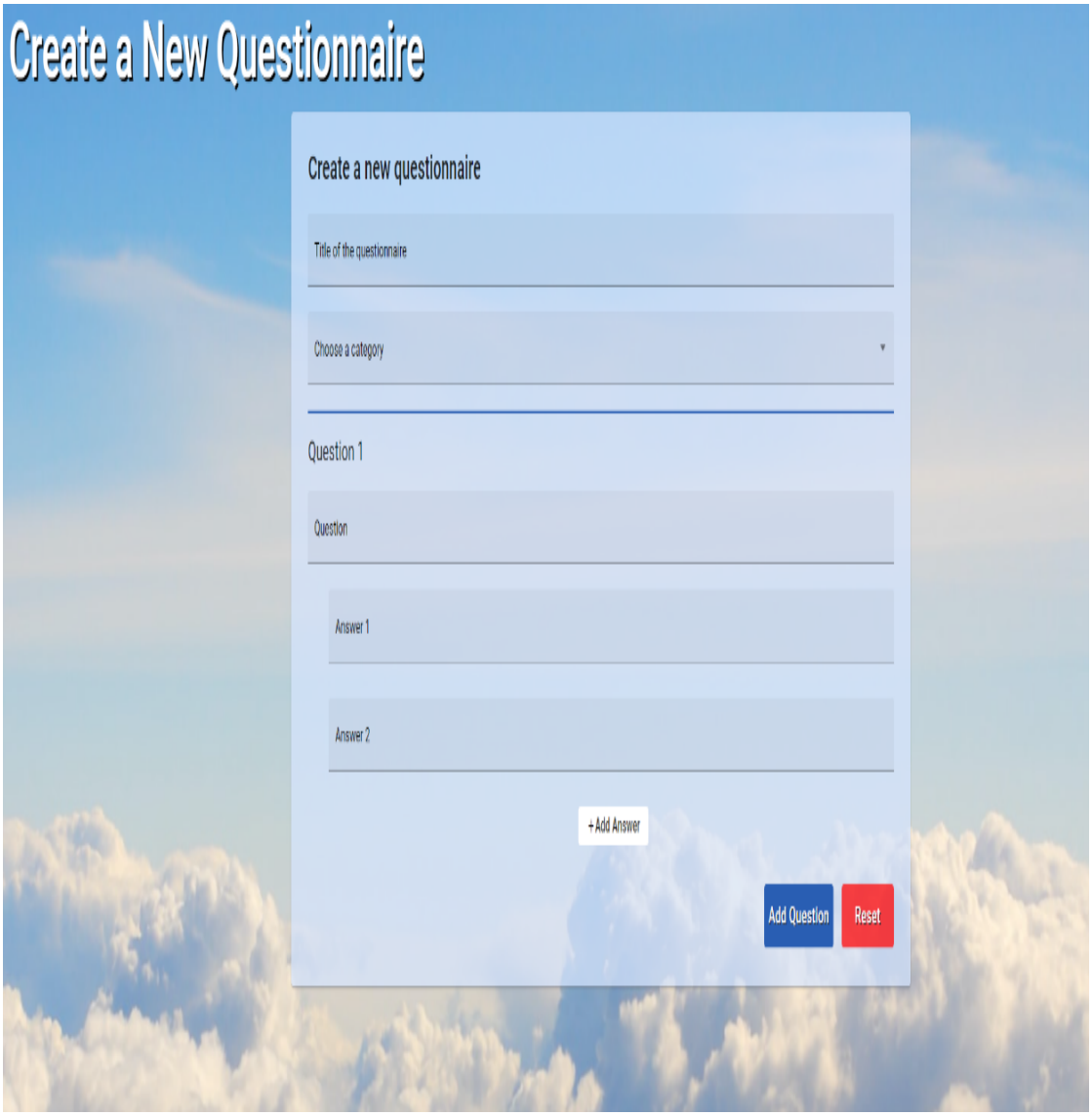

Fig. 4: New Questionnaire

From the questionnaire list you can access the questionnaire share link, you can delete one and you can see it's details containing statistics and responses.

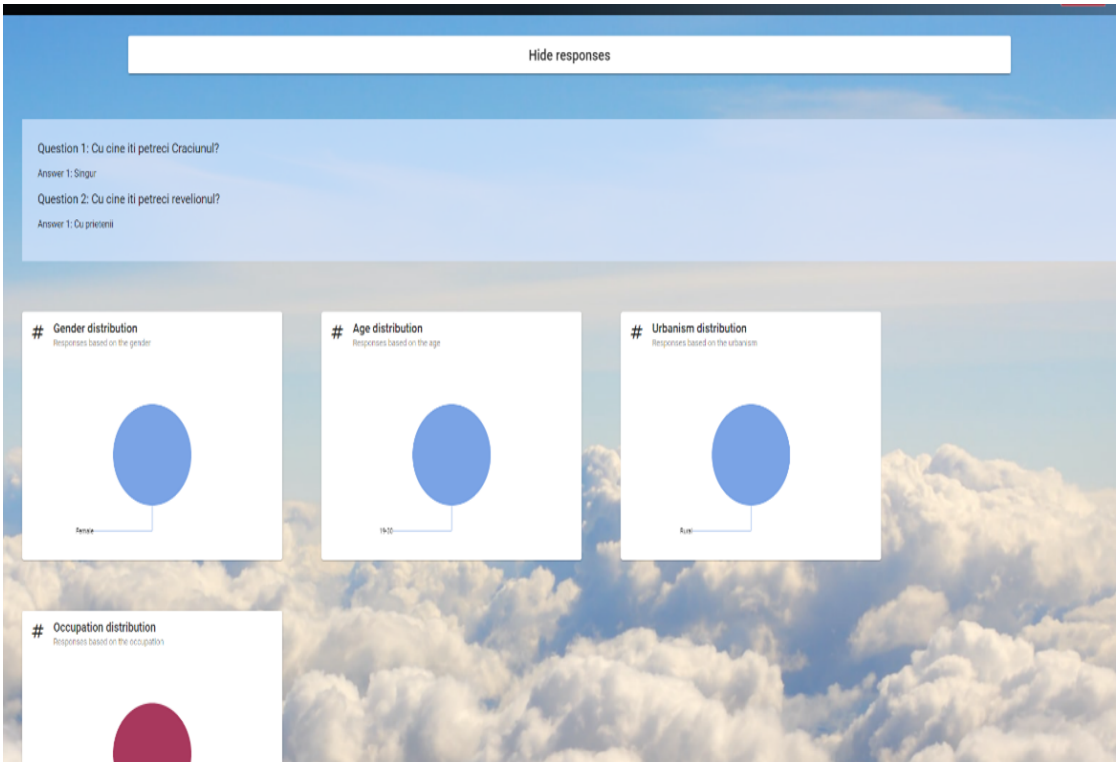

Fig. 5: Statistics

Backend

Installation and configuration

Since the backend is a node-express-mongo application, the following requirements need to be completed in order for the application to work properly:

- For mongo installation the next link was used:https://docs.mongodb.com/manual/tutorial/install-mongodb-on-windows/
- after the mongo is installed, an account for a cluster needs to be created in order to use your own mongodb instance
- after the mogno part is ready, the next commands need to be introduced in the terminal in this order: cd ../workdirectory, npm i, node run watch:server
- after all the commands are written, the server will be hosted on localhost:3000 (this can also be changed)

Structure of the backend

The backend is split into the next parts:

- *Models* represents the blueprint of the documents that are saved in the Mongodb
- *Lib* is the last layer of communication between the backend and the database, all the CRUD operations are prepared in the Lib
- *API* represents the second layer where the data is transmitted from the request to the lib
- *Middleware* represents the first layer of communication where the request is intercepted or transmitted to/from the frontend
- *Routes* represents the actual url's that have the middleware functions attached

Endpoints of the backend

The backend has the following endpoints:

- *POST* /register/user that containts all the information of the users, if the user wants to be anonym, the username will be generated and the email will be the same as the username. If the user does not provide a username or an email, the other will be duplicated
- *POST* /login/user that append the user if the acces is granted or not token
- *GET* /getUsers is used for admins to see the users that use the app
- *POST* /getUser with the body of either the username or the email in order for the admins to find a specific user
- *GET* /category/one/category/categoryName is used by the admin to see a specific category
- *GET* /category/all is used to see all the categories
- *POST* /questionary/create/categoryName is used for by a user to create a survey that is coupled to the username
- *GET* /questionary/questionaryName||questionaryCode is used by the admins in order to visualize the characteristics of the survey
- *GET* /questionarycat/categoryName is used to display all the surveys on specific categories
- *DELETE* /questionary/username/questionaryName||questionaryCode is used by a user to delete the surveys
- *GET* /questionaries/username is used by a user to see all of his/her surveys and also see how many uniqueAnswers each survey has
- *GET* /questionary/answer/unique/questionaryName is used by a user in order to see how many unique answers a specific survey has

- *GET* /questionary/get/statistics/questionaryName is an aggregation of piece of information that shows the statistics of users' details for a specific survey
- *POST* /questions/create/questionaryName is the endpoint that creates the answer with the list of possible answers
- *GET* /questions/questionaryName||questionaryCode is used in order to see all the questions of a specific survey
- *GET* /oneQuestion is used in order to find a specific question based on its name
- *DELETE* /question is used in order to delete a specific question
- *POST* /answer/create is used by the user to send the chunk of questions and also the personal details, even if he or she logs in as an anonymous user
- *GET* /answer/questionID/questonaryCode is used in order to see all the answers of a survey with the personal data of the users

Future endpoints that are not yet displayed in the Graphical interface:

- *POST* /category/create is used by a user to create a specific category
- *POST* /questionary/anonym/create is used by a user to create a anonymous survey, it will have attached a generated code in order for the user to send the survey via code

Bibliography