# Design documentation

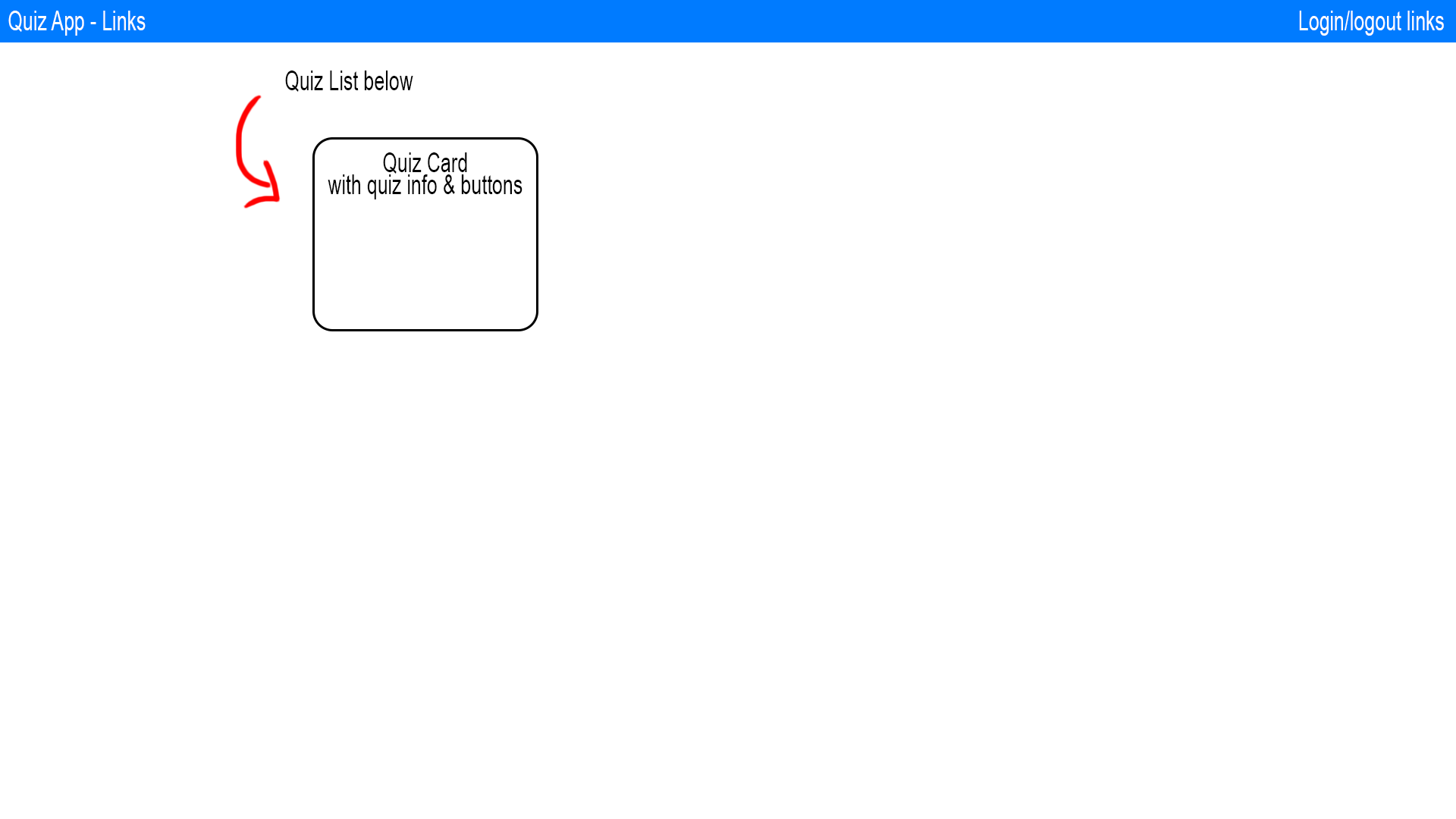
## Requirements

Project requirements:

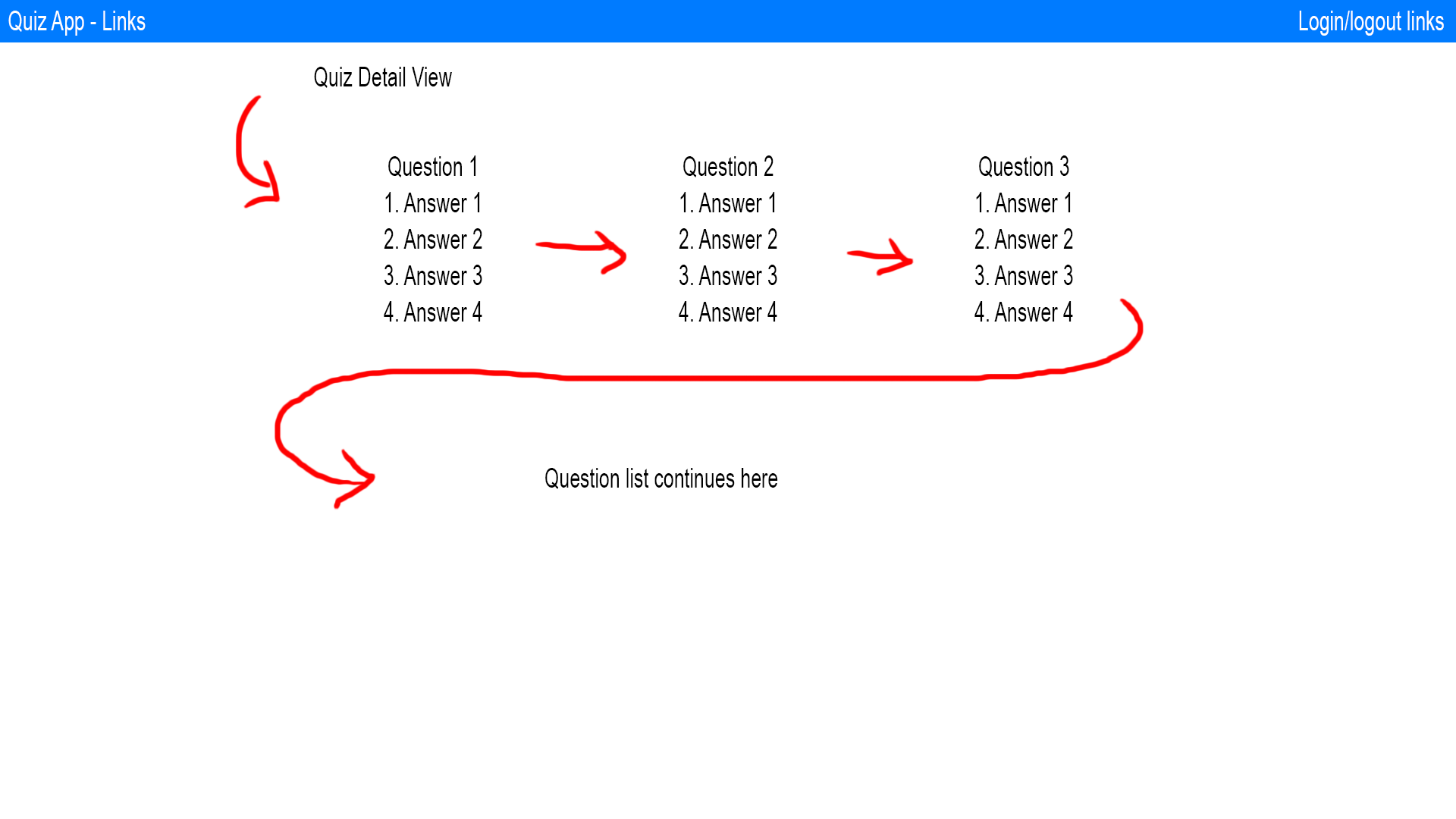
1. Python 3.9.5 with Django framework – back-end development.
2. Django crispy forms – back-end third-party package (makes Bootstrap-friendly forms).
3. HTML, CSS, JavaScript – front-end
4. Bootstrap4 – front-end framework for responsive & mobile-friendly front-end designs.
5. jQuery – required for some Bootstrap4 modules to work.
6. FontAwesome – a set of icons.
7. Flake8 – to check my code against coding styles(PEP8), programming errors and to check cyclomatic complexity.

## UI sketches (main pages)

Quiz Homepage. It contains a list of quizzes. Each quiz is grouped into a card (Bootstrap – Cards). You get redirected to quiz detail page after clicking on quiz.



Quiz Detail page. It displays a list of questions (and answers if permissions allow to do so). If authorised user is allowed to edit – it will display additional buttons to add, edit and delete questions.



## System structure

The system contains the following pages:

1. Homepage – displays a list of quizzes. Allows user to CRUD quizzes depending on permission level.
2. Quiz Detail page – Each quiz has a detail page (created dynamically). This page displays a list of questions (and answers depending on request user perm level). Allows user with edit perms to add/edit/delete questions.
3. Creation pages for quizzes, questions & answers (question and answers and grouped into one creation page using Django formset factories).
4. Quiz Start page – Each quiz also has a quiz start page. It displays quiz in a way where each question can be answered. Upon completion user gets redirected to the results page.
5. Login page – Speaks for itself. Allows you to enter username and password to log in. Registration page is not implemented.

## Model diagrams

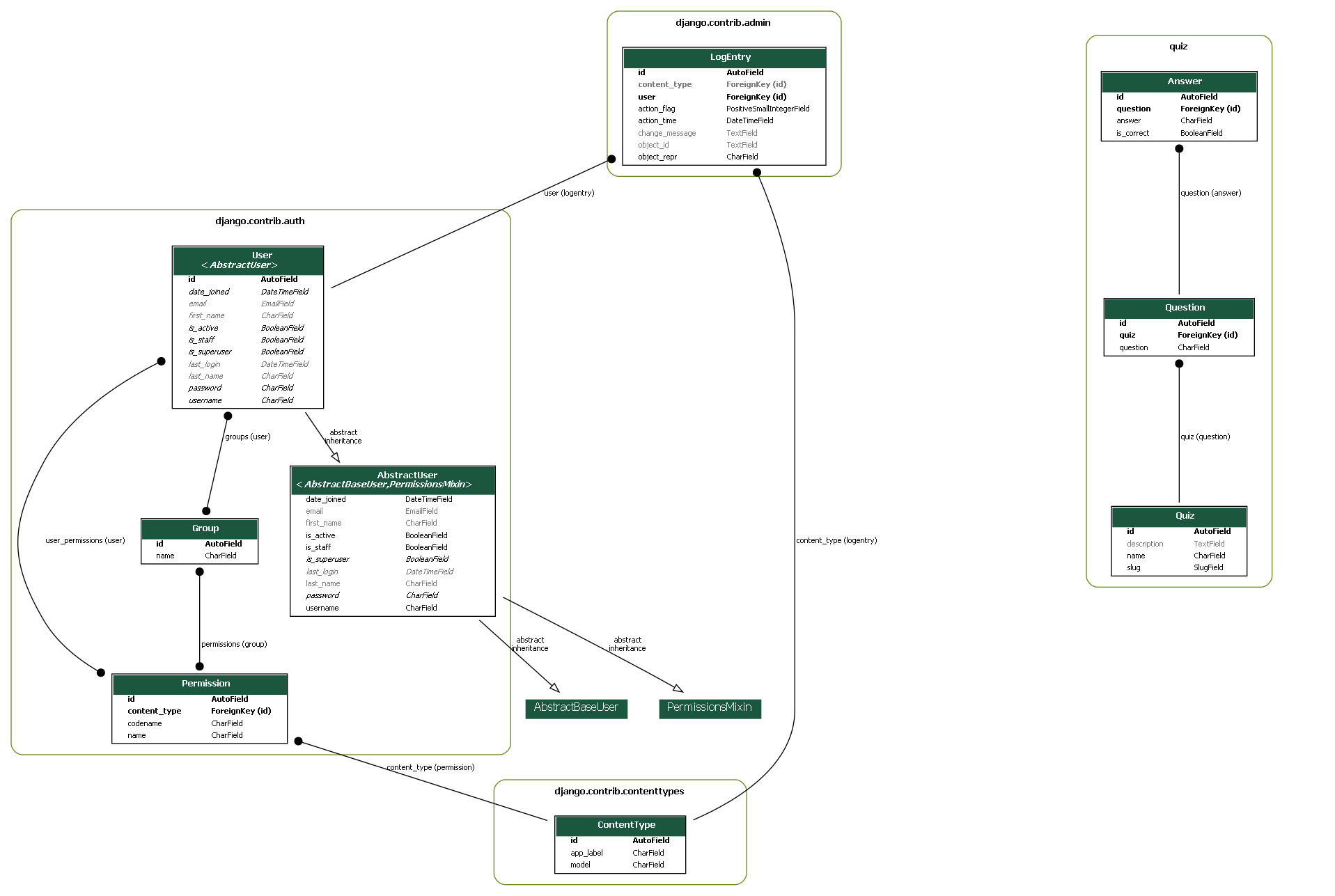
The diagram below contains both quiz and user models. I use default Django Auth user models (left diagram).

Each user has three permission levels defined by two Boolean fields:

1. Restricted – is\_staff & is\_superuser set to False.
2. View – is\_staff = True, is\_superuser = False.
3. Edit – is\_staff & is\_superuser set to True.

Quiz app functionality is defined by three models:

1. Quiz model. Contains four fields:
   1. ID – Django AutoField.
   2. Name – Displayed name of the quiz.
   3. Description – Displayed description of the quiz.
   4. Slug – slugified version of name field for user-friendly url paths (testwebsite.com/quiz/test-quiz-123/edit/ instead of testwebsite.com/quiz/1/edit/).
2. Question model. Contains three fields:
   1. ID – Django AutoField.
   2. Quiz – Foreign key to Quiz model (1-many relation).
   3. Question – Displayed question.
3. Answer model. Contains four fields:
   1. ID – Django AutoField.
   2. Question - Foreign key to Question model (1-many relation).
   3. Answer – Displayed answer.
   4. Is correct – Boolean value to define correct answer.

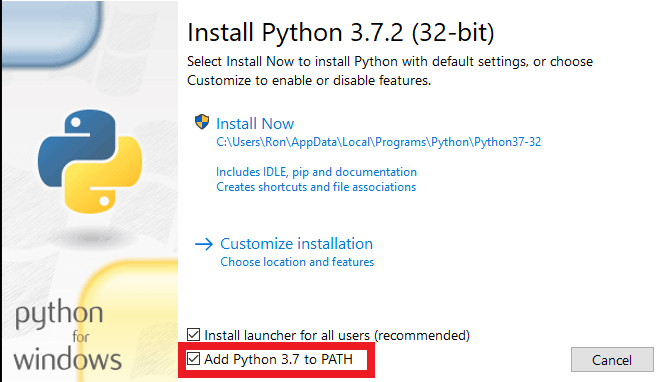


# User guide

## project launch guide

Python 3 required. <https://www.python.org/downloads/>. I use Python 3.9.5 in this project. Make sure to check “Add Python to PATH” if installing on Windows.

Example (screenshot from google search):



1. Unpack zip folder to desired location (exeter-college-quiz-final).
2. Launch console/cmd.
3. Navigate to folder using console/cmd.
4. Install virtualenv with “pip install virtualenv” in console.
5. Launch existing “.venv\_project” using:
   1. “source .venv\_project/bin/activate” on Linux.
   2. “.venv\_project/Scripts/activate” on Windows.
6. Run the project by typing “python manage.py runserver” in the console.
7. It can now be accessed on <http://127.0.0.1:8000/> in your browser.

## Login details (for each permission)

### Edit

LOGIN – edit

PASSWORD – 1234

### VIEW

LOGIN – view

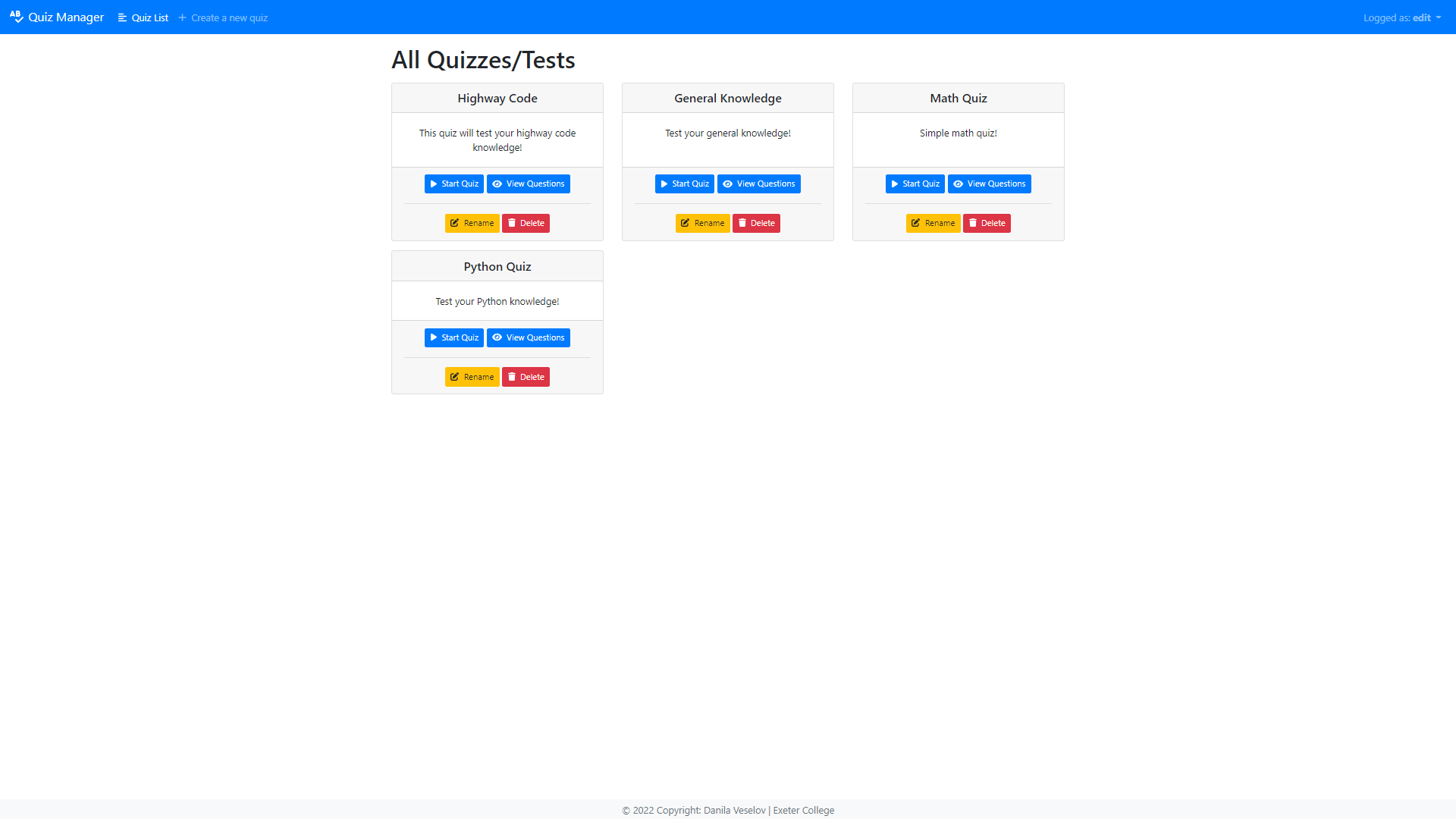
PASSWORD – 1234

### Restricted

LOGIN – restricted

PASSWORD - 1234

## Homepage



Dropdown. Displays your username.

Allows to logout.

Quiz List – takes user to quiz list page.

Create a new quiz – takes user to quiz create page

Start quiz – takes you to quiz-start page (quiz completion page).

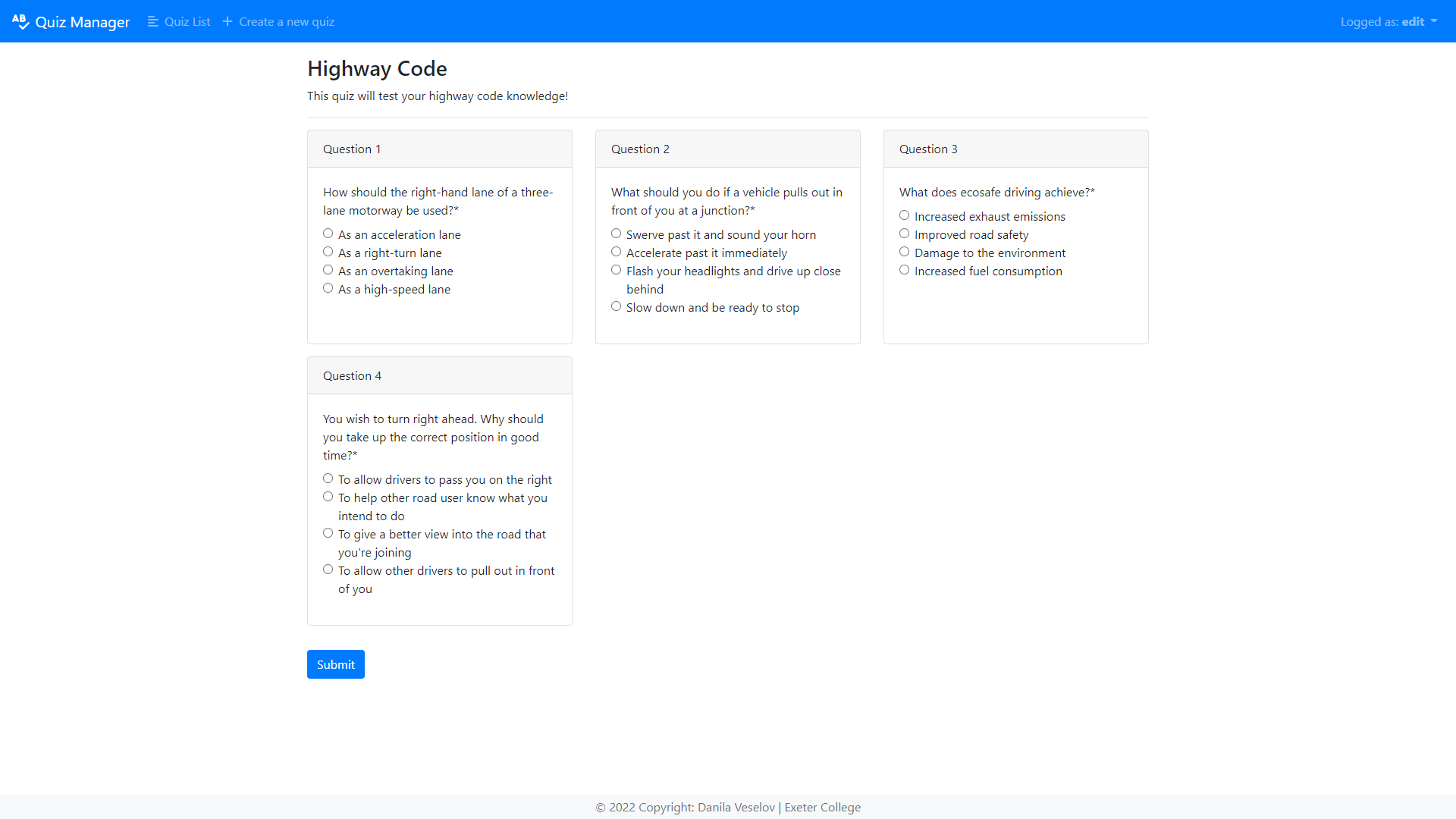
View Questions – allows you to CRUD questions depending on permission level.

Rename – allows you to rename the quiz.

Delete – deletes selected quiz.

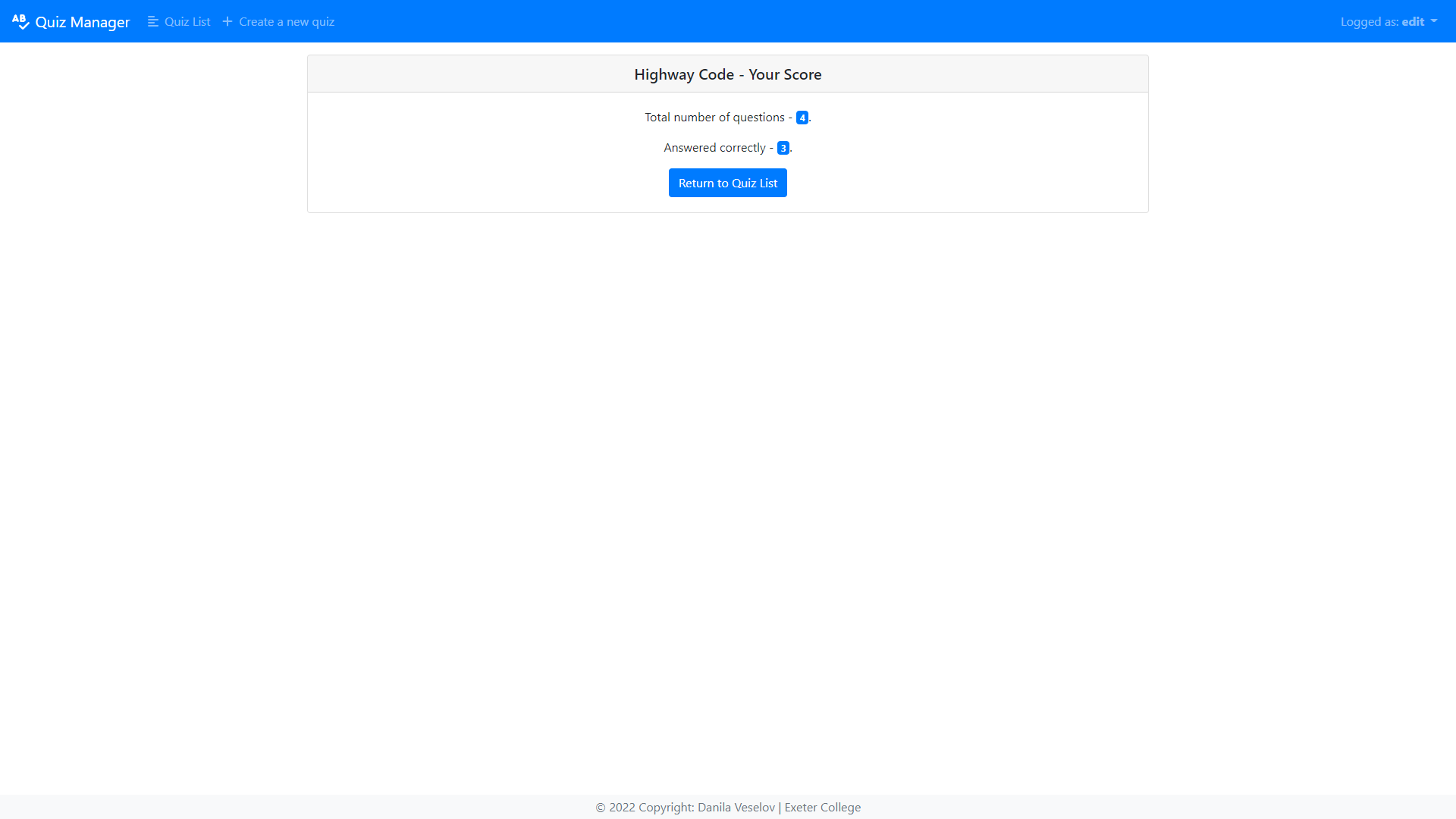
## Quiz start page

You can undertake selected quiz on this page.



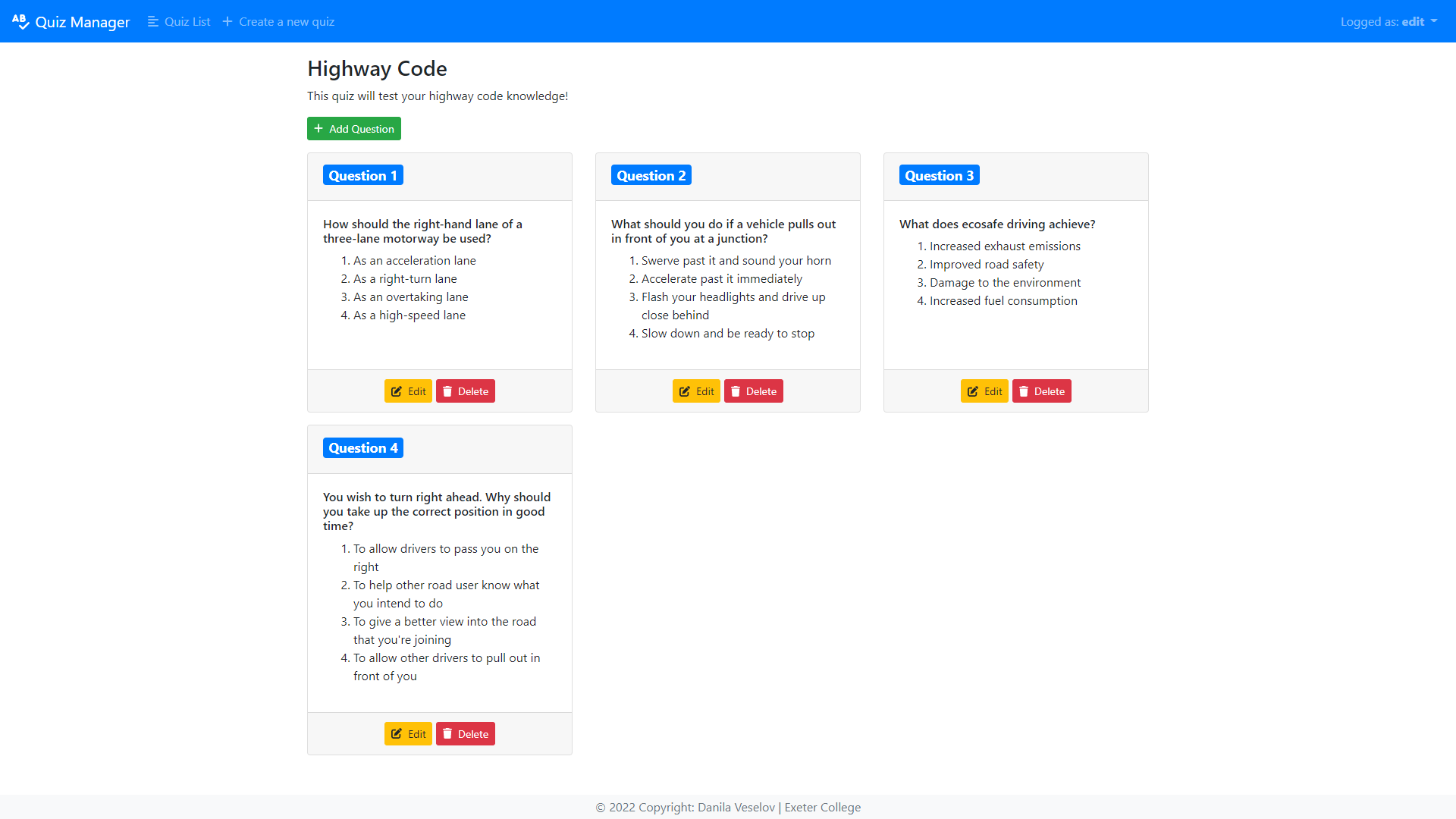
## Quiz results page

This page gives you results based on your choices.



## Quiz detail page

This page allows you to add questions to the quiz and edit/delete existing questions.

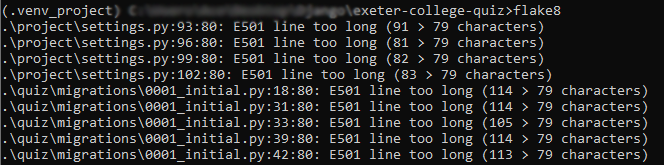


# Testing

## Flake8 testing

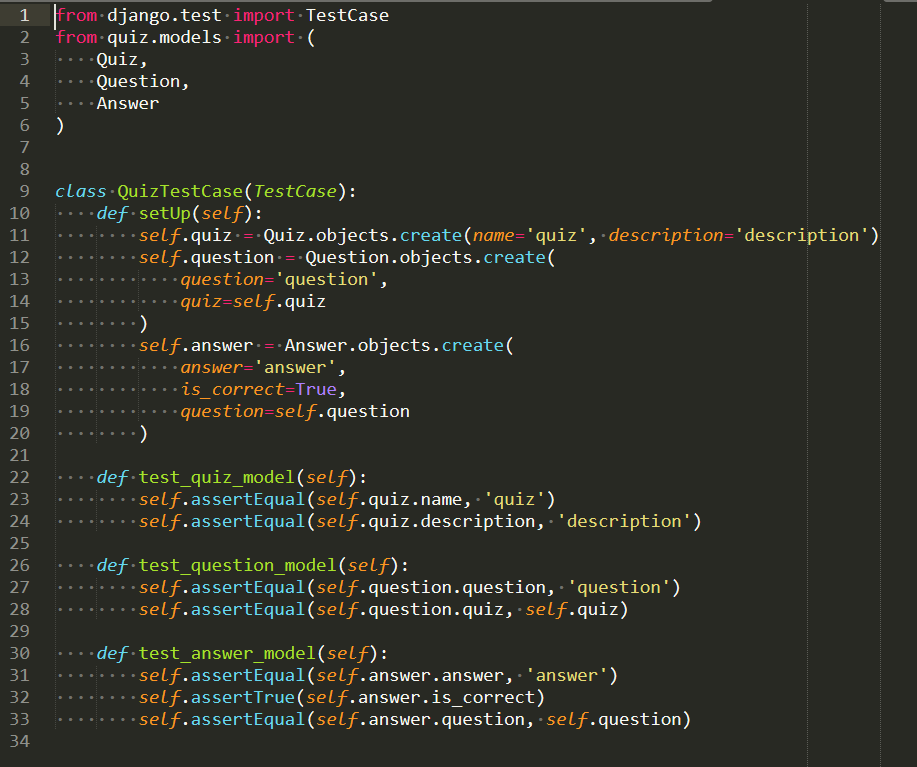
I use flake8 for testing in this project. It allows me to check my code against coding styles(PEP8) and programming errors. It can be executed by running flake8 in the console.

Example:

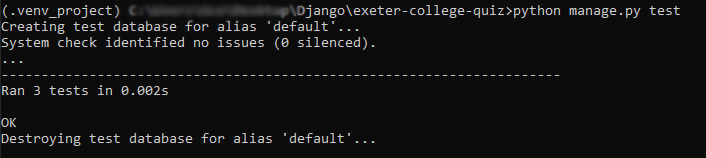


## Unit tests

I also have created a unit test for each model. Tests create a test object for each model in test database and then compares the created record with static data.



They can be executed by typing “python manage.py test” in the console.



# Conclusion

The project went quite well from my point of view. I managed implement all the desired login. However, there is still plenty of aspects that can be improved and at least one bug to be fixed.

## Limitations & Suggestions

There is a bug that does not let you get the results of a quiz if the selected quiz contains a question with no associated answers. It can be easily fixed by adding an if or try statement, but I was running out of time and did not manage to fix it.

Front end can be definitely improved too. I currently have a lot of white space that can be filled.

Quizzes should have tags/categories and search for easier navigation, but I didn’t have enough time to implement this functionality.

Quizzes should also have a “publish” boolean field to allow publish/unpublish functionality (without having to delete them).