**Testing Site Project – Ofek Shaked**

# Main Objectives:

* Create option for multiple organizations.
* Create option for multiple topics for each organization.
* Create tests for each subject.
* Create questions, each question can be related to more than one subject and test.
* Create new test object for every person who took the test with their answers and details.
* Create an option to get reports for either tests or students, students are not joined together(they are created each time as new student for each test).
* Create a testing engine for students to get tested on based on test unique id.
* Update tests and modify them.
* Update questions and modify them.

# Time Plan

Time plan for the project:

* Coding the backend up till tests taken – should take about a day
* Starting building the front-end structure – day or two
* Testing backend after adjusting it if needed – day or less
* Connecting frontend structure to the backend – should take about two days
* Testing frontend - day
* Adding backend code for tests taken - half a day
* Adding structure for adding and updating tests – should take between 1-2 days
* Testing functionality – day or two
* Adding student testing components – about a day
* Testing all – 2 days
* Code review – 2-3 days

# Models

* Organization
  + Id – objectId/text
  + Name – text
* Topic
  + Id – objectId/text
  + OrganizationId - objectId/text
  + Name - text
* Question
  + Id – objectId/text
  + Type – text but based on enum like type
    - SignleChoiceQuestion
    - MultipleChoiceQuestion
  + Text – text
  + InnerText – text
  + Orientation
    - Horizontal
    - Vertical
  + OptionalAnswers – array of type [Answer](#_Models)
  + Tags – array of text
* Answer
  + Id – objectId/text
  + Text – text
  + IsCorrect – boolean
* Test
  + Id – objectId/text
  + TopicId - objectId/text
  + Language – text based on enums
    - Hebrew
    - English
  + Name – text (max 200 length)
  + Instructions – text
  + IsAnswersShown – boolean
  + DiplomaUrl - text
  + SuccessText – text
  + FailedText – text
  + Questions – array of question ids
* User
  + Id – objectId/text
  + FirstName – text
  + LastName – text
  + Email – text
* Test\_taken
  + Id – objectId/text
  + Test\_id – id of test taken
  + Test\_taken\_questions – array of ids of the questions in right order
  + User\_id – object of user
* Test\_Taken\_question
  + Id – objectId/text
  + Test\_taken\_id – id of test taken
  + Question\_id – id of question
  + Answers\_chosen – array of answer ids

# Backend

## Organization Route

* Get all organizations /organization get

## Subject Route

* Get all subjects /subject get

## Test Route

* Get All Tests /test get
* Add Test /test post
* Update Test /test/id post
* Get test by params /test&params

## Question Route

* Get all questions /question get
* Add question /question post
* Update question /question/id post

## Tests Taken Route

* Add test taken /testtaken post
* Get tests by parameters /testtaken&params get

# Frontend

## Src

### Components

#### Common

##### Table component

Shared table to keep the style the same all around

#### Home

##### Home component

Default layout of the webapp contains the whole app

#### Pages

##### Main

###### MainPage

Main page of the app as know as the landing page

##### Questions

All pages that relate to questions

###### Add/Update question

Page for adding or updating a question

Answers

Components related to question's answers

###### Modify questions

Page for showing all of the questions in a table and performing actions on them

###### Question

Page for showing a question as a user

#### Routers

##### Main Router

Main router of the app leading to pages

### Hooks

### App