PROJECT REPORT ON

FeeCo - Next Gen Feedback Collection Portal

Submitted by:

Swati Kanchan

Roll no: 15/CS/95

Under the Guidance of:

Dr. Debasis Mitra

May, 2019

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY, DURGAPUR

WEST BENGAL - 713209

DECLARATION

I hereby declare that this submission is my own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degrees or diplomas of the university or other institutes of higher learning, except which due acknowledgment has been made in this text.

Date:

Swati Kanchan

15/CS/95

CERTIFICATE

This is to certify that Miss Swati Kanchan (Roll no. 15/CS/95) have completed the project titled “FeeCo - The Next Gen Feedback Collection Portal” at National Institute of Technology, Durgapur under my supervision and guidance in the fulfillment of requirements of Eighth Semester, Bachelor of Technology (Computer Science and Engineering).

Date:

____

Dr. Debasis Mitra

Associate Professor

Department of Computer Science and Engineering

Date:

____

Dr. Tandra Pal

Professor and Head of the Department

Department of Computer Science and Engineering

ACKNOWLEDGMENT

It is a genuine pleasure to express my deep sense of thanks and gratitude to my teacher, mentor and project guide Dr. Debasis Mitra, Associate Professor, Department of Computer Science and Engineering, NIT Durgapur for selecting me for this awesome Feedback Collection and Summarizer project and guiding me throughout. Without his active support, help and cooperation I would not have complete this project.

I would also like to extend my gratitude to Prof. Anupam Basu, Director of NIT Durgapur for providing me with all the facilities that was required.

I am extremely thankful to my parents and members of my family who supported and cared for me morally as well as economically.

I am also thankful to one of my friends and classmates Lokesh Nandanwar for providing me necessary technical suggestions during the designing of the website.

Place: NIT Durgapur

Date:

____

Swati Kanchan

15/CS/95

swatikanchan707@gmail.com

ABSTRACT

Feedbacks provide valuable information that will be used to make important decisions for the future. The most effective leaders actively seek feedback to enhance their performance. It helps learners to maximize their potential at different stages of training, raise their awareness of strengths and areas for improvement, and identify actions to be taken to improve performance. The feedback form is used for any type of user, clients, customer to rate an event and provide feedback by using this form. One such type of use case involves teachers and students, where teachers take feedbacks from their students about their experience during his/her teaching practice.

This is a project in which a hassle-free platform is created for generating feedback forms and sending those form links to people along with a One-Time-Password (OTP) via mails. The form receivers can then fill those forms by clicking on the link provided and using the OTP. Form filling can be done anonymously or non-anonymously as per the wish of the form creator. The latter can also download the responses got from the users in '.csv' format. Along with this, an automatic feature is added which summarizes the feedback comments (which is in the form of text) for the feedback creator. For this project, the person who is creating or generating several forms is represented by a "Teacher" and the ones who are filling the forms are represented by "Students.

CONTENTS

Introduction

What is a survey? It seems like a simple question, but as with many things, the answer is more complex than many people appreciate. Surveys can take multiple forms but are most common in the form of a questionnaire, either written or online [1]. Fundamentally, a survey is a method of gathering information from a sample of people, traditionally with the intention of generalizing the results to a larger population. Surveys provide a critical source of data and insights for nearly everyone engaged in the information economy, from businesses and the media to government and academics. Millions of surveys are sent out each year and although some companies still send out paper surveys, there would be a lot of wasted paper if most were not digital. Online survey software has been the most popular way of conducting survey research for over a decade now, and because getting faster insights is imperative to business success, more and more companies will migrate to digital solutions.

It's easy to create a survey with digital software, plus, digital software can save your company time and money because they have lower setup and administrative costs. It's more convenient for the customer or respondent because they can take the survey on whichever digital device is most convenient for them (tablet, computer, phone, etc). It's also more convenient for you because you just need to send the survey link via email and you'll have the data in your survey management software as soon as responses come in. Digital surveys scale faster. With the click of a button in the survey tool, you can send a survey to thousands of people and even translate it into multiple languages. You can also analyze the results easily, and they're more accurate because you don't have humans putting the data into a computer to be analyzed. There are many things you can do with your survey results, but it's most important to get them into the hands of the decision makers in an easily understandable format. From creating interactive PDF reports to exporting to Excel, or Google Sheets, you can present the information in multiple ways. In this era of Artificial Intelligence, it is the necessity of the survey software to have Automatic summarization of the feedbacks or responses from the respondents. This project aims to cover all the shortcomings of the free and open source survey software we have in the online market.

Objective

This project aims to develop a Online Survey Tool **FeeCo**, free and open source online Feedback Collector tool written in Node and Angular based on a MongoDB database system. As a web server-based software it enables users using a web interface to develop and publish online surveys, collect responses, summarizes the responses, create statistics, and download the resulting data.

FeeCo is a web application that is installed to the user's server. After installation users can manage FeeCo from a web-interface. Users have Once a survey is finalized, the user can activate it, making it available for respondents to view and answer. Likewise, questions can also be imported and exported through the editor interface. FeeCo has no limit on the number of surveys a user can create, nor is there a limit on how many participants can respond. Aside from technical and practical constraints, there is also no limit on the number of questions each survey may have.

FeeCo also provides basic statistical and graphical analysis of survey results. Surveys can either be publicly accessible or be strictly controlled through the use of "OTP" tokens, granted only to selected participants. Additionally, participants can be anonymous and feedbacks can be summarised from the responses.

Literature Review

- ***Survey Monkey:***

  SurveyMonkey [2] is an online survey development cloud-based software as a service company. It was founded in 1999 by Ryan Finley and Chris Finley. It allows users to create their own surveys using question format templates. The basic version of SurveyMonkey is free; an **enhanced** version is also available at a cost. It offers a free account (Basic plan) and three paid options. With their **free** Basic plan, we can create and send a survey with up to 10 questions or elements (including question types, descriptive text, or images) within a matter of minutes and view up to 100 responses per survey. The main difference between the plans is the number of questions and responses allowed for each survey, as well as the team collaboration features.

- ***Lime Survey:***

  LimeSurvey [3] is a free and open source online statistical survey web app written in PHP. Whether you are conducting simple questionnaires with just a couple of questions or advanced assessments with conditionals and quota management, LimeSurvey has got you covered. Also, it has its limitations when it comes to some complex question types, as it has its own set of default questions and other platforms might have a larger variety of questions to use, which might save time when creating a survey. It is an excellent platform and beats SurveyMonkey and all other platforms in terms of features and ease of use.

Software Requirements

- ***System requirements:***

  Server installed with NodeJS: This application's backend website needs to be deployed on a server which needs to be installed with NodeJS from where all the APIs would be accessible for the end-users computer or mobile.

- ***Technology Stack:***
  - NodeJS v10.9.0 [4]
  - ExpressJS ^4.16.4 [5]
  - Angular v6.1.3 [6]
  - MongoDB v4.0.5 [7]
  - MongooseJS ^5.4.12 [8]
  - Sumy [9]
  - Python 3 [10]

Project Plan

- ***Whole architecture:***

  The whole architecture of this application development project is shown in (Fig. 1). There are 2 servers installed in the cloud, one for the backend APIs and feedback summariser, and another for the MongoDB database. The client or the frontend can then be accessed through the hosted website's domain name or IP address.

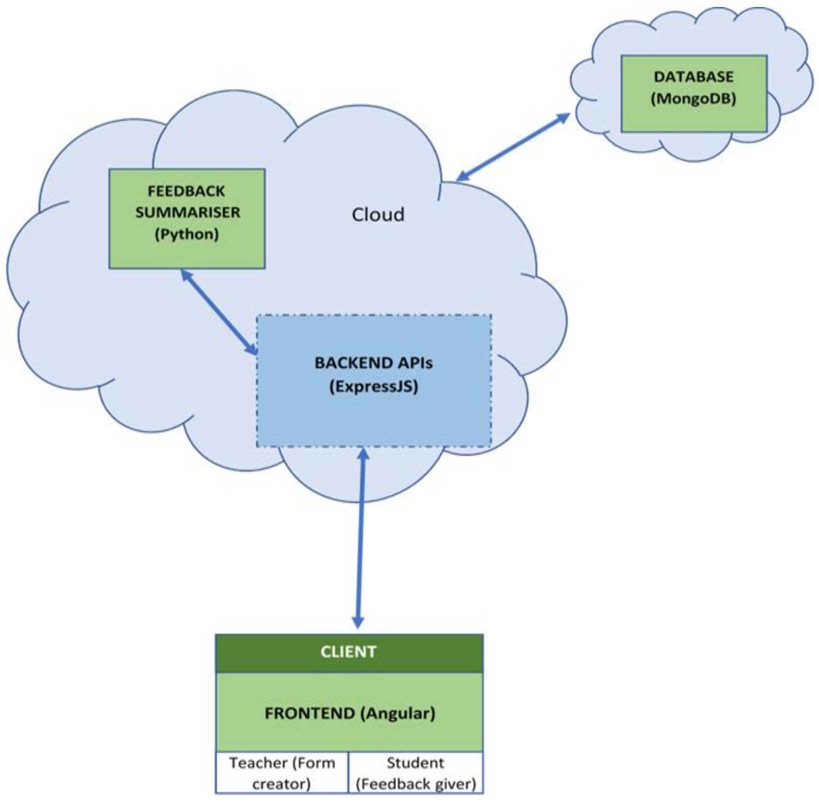


Fig 1: Whole Architecture of the Web Application

- ***Frontend:***

  The frontend contains 2 auxiliary and 2 main pages:
  - Auxiliary
    - Signin/Signup page: For every Teacher/Form Creator
    - Response page: For every Form
  - Main

        - Home page: This is where a Teacher sees all his/her created forms so far and also some more features of creating new forms.
        - Form Creation page: This is for every form, where the forms can be edited, new questions can be added and advanced settings of the form can be accessed such as activating/deactivating form collection, sending form links via emails and getting the form link to be copied to clipboard.
        - We have used Angular framework to make the frontend. Services are used to integrate the backend with it.

- ***Backend:***

  The backend contains several API routes written in ExpressJS which are used by angular services to fetch and store data in the database and get results from our Machine Learning model, feedback-summarizer. (Fig. 2)

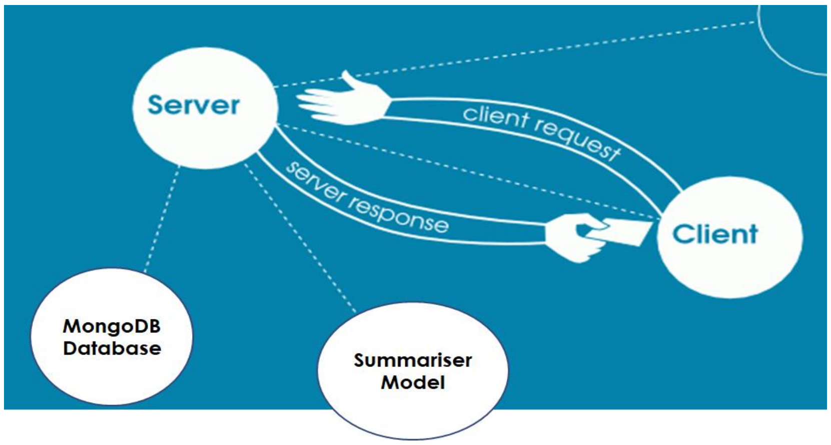


Fig 2: Working of backend

- ***Database Design:***

  MongoDB, a NO-SQL database, is used for our database system. It will be hosted in mLabs cloud server in future. 3 to 4 collections or tables are created as of now whose detailed implementation details are specified in next chapter. As the project's goal intensifies, this number can increase in future. Following are the names and further description of documents or attributes of the tables (Fig. 3):

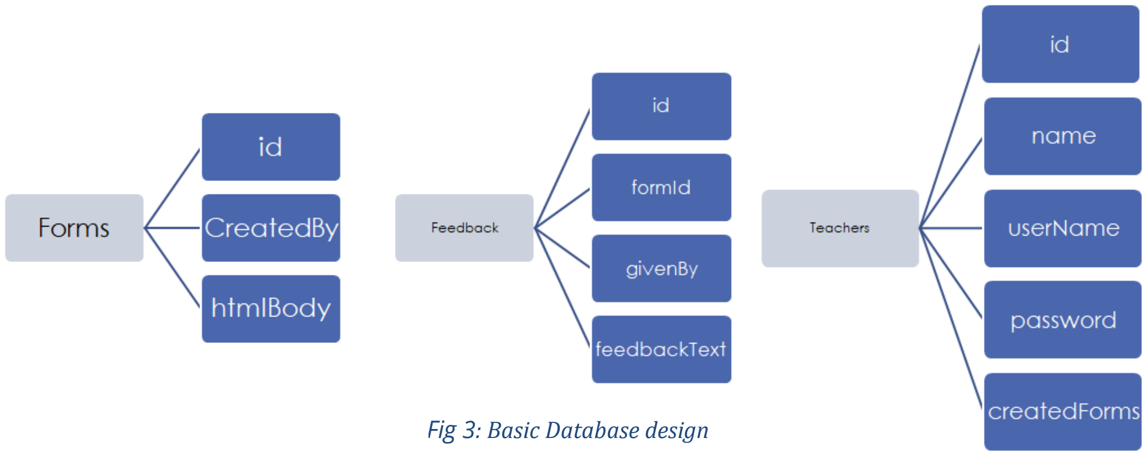


Fig 3: Basic Database design

- ***Text summarizer:***

Machine learning (ML) is a category of algorithm that allows software applications to become more accurate in predicting outcomes without being explicitly programmed.

We have used Extractive Text Summarization Techniques with Sumy which performs summarization by picking portions of texts and constructing a summary [11]. (Fig. 4)

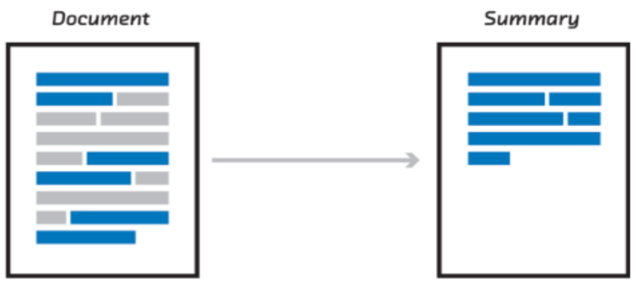


Fig 4: Text summariser simplified

This is our pre-trained Machine Learning model which is stored in the same cloud where our website is hosted. It summarizes the Feedback texts received from every respondent automatically using the pre-trained weights.

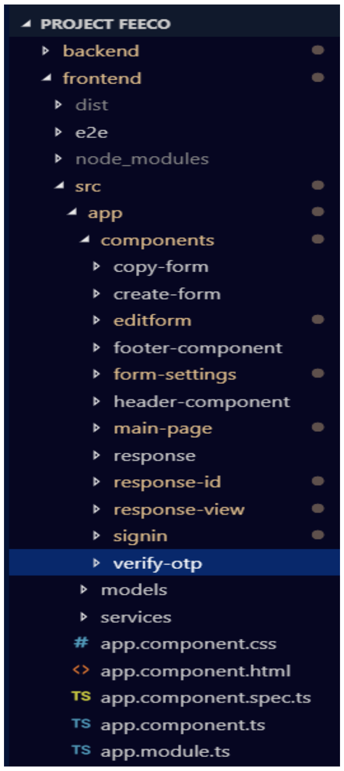
Design Strategy and Implementation

➢ Frontend

The frontend of the application is developed using Angular 6. It is installed by installing NodeJS from the official website [4] and then installing Angular CLI using the commands:

npm i -g typescript
npm i -g @angular/cli

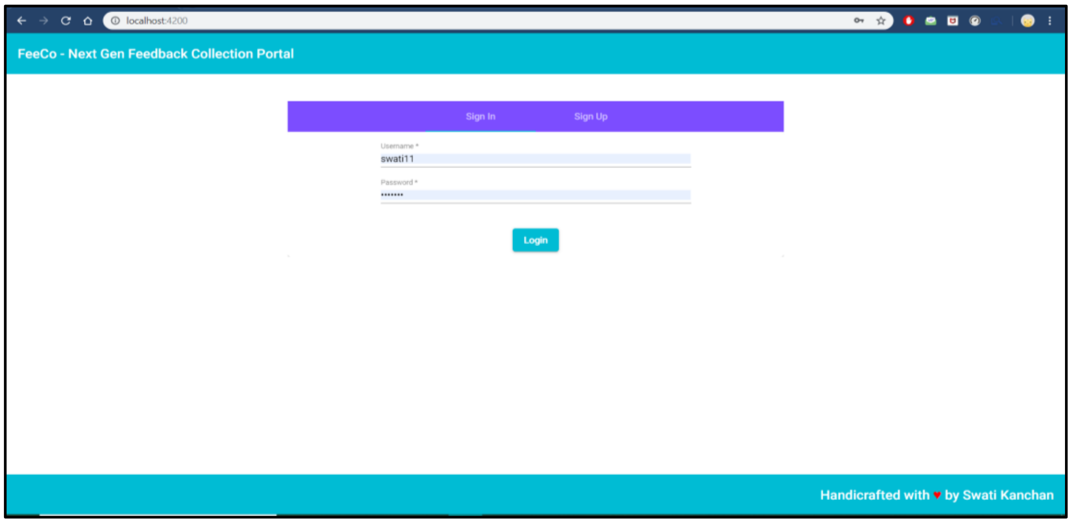
We have created a new angular project named "frontend" by using the command ng **new** frontend inside our main project "PROJECT FEECO". The following features are developed by creating several components in the angular project each with its separate Typescript, HTML, and CSS file. The main components created are as shown here (Fig. 5). The root component where all other child components are nested is known as "app component".



```
PROJECT FEECO
  backend
  frontend
    dist
    e2e
    node_modules
    src
      app
        components
          copy-form
          create-form
          editform
          footer-component
          form-settings
          header-component
          main-page
          response
          response-id
          response-view
          signin
          verify-otp
        models
        services
        app.component.css
        app.component.html
        app.component.spec.ts
        app.component.ts
        app.module.ts
```

Fig 5: Components of Angular Project

1. Sign-in and Sign-up page



localhost:4200

FeeCo - Next Gen Feedback Collection Portal

Sign In | Sign Up

Username *
swati11

Password *
•••••••

Login

Handicrafted with ♥ by Swati Kanchan

Fig 6: Sign-in and Sign-up Page

This page contains 2 tabs for signing in to the application by an existing user and for signing up into the application as a new user by filling their personal details. (Fig. 6)

a. ***Personal Details in Sign-up Form:***

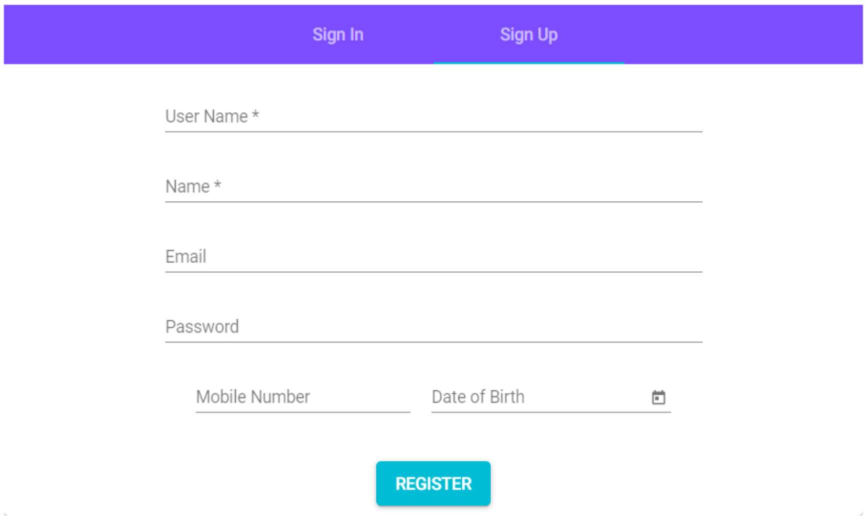


Fig 7: Fields in Sign-up page

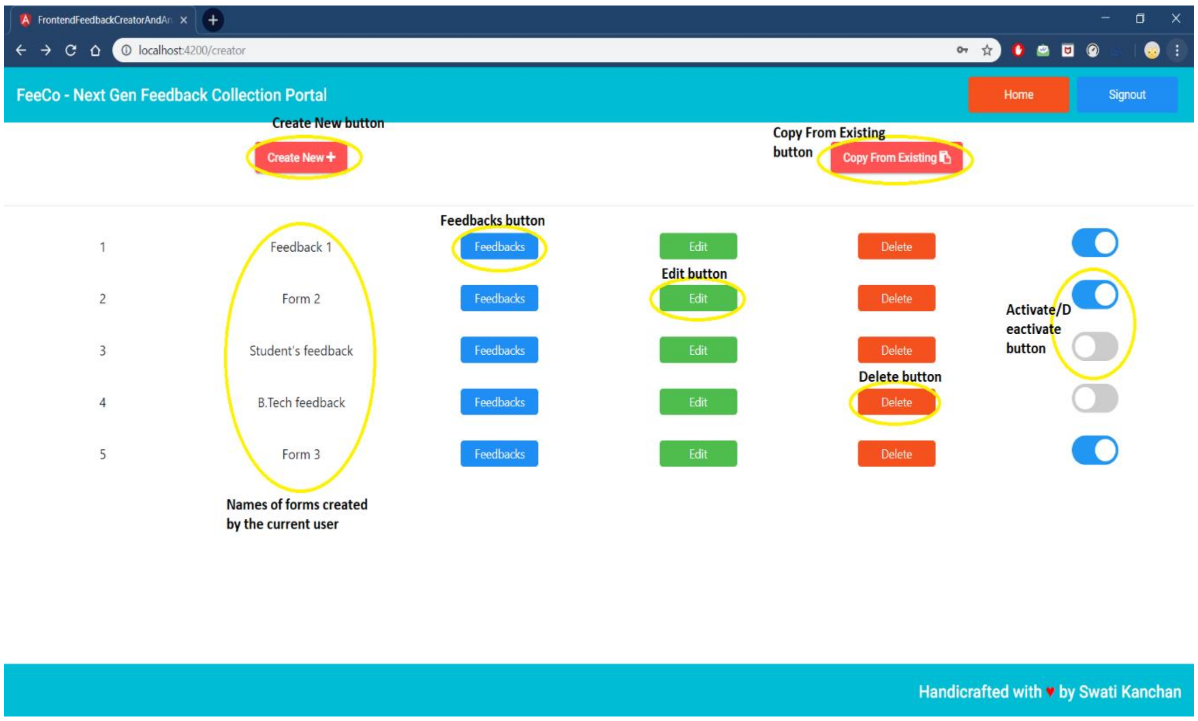
Various fields required to be filled up to register a new user are Username, Name (to be displayed to other users), Email id, Password, Mobile number and Date of Birth. (Fig. 7)

b. ***Password setup and login***

User has to create a password while signing up and use it to sign in later. Password will be stored in the database in the encrypted form for security purposes.

2. User Dashboard

A user dashboard is a user interface that, somewhat resembling an home screen of the application, organizes and presents information in a way that is easy to read. However, a computer dashboard is more likely to be interactive. To some extent, most graphical user interfaces (GUIs) resemble a dashboard. We will be using this users dashboard for carrying the most common tasks. A sample dashboard of a user is shown below and the functionalities present in the dashboard are also listed below (Fig. 8).


FrontendFeedbackCreatorAndAn
localhost:4200/creator
FeeCo - Next Gen Feedback Collection Portal
Home
Signout
Create New button
Create New +
Copy From Existing button
Copy From Existing
Feedbacks button
Edit button
Delete button
Activate/Deactivate button
Names of forms created by the current user
Handicrafted with ♥ by Swati Kanchan

1	Feedback 1	Feedbacks	Edit	Delete	●
2	Form 2	Feedbacks	Edit	Delete	●
3	Student's feedback	Feedbacks	Edit	Delete	○
4	B.Tech feedback	Feedbacks	Edit	Delete	○
5	Form 3	Feedbacks	Edit	Delete	●

Fig 8: User Dashboard

a. *View all forms*

This view will show all the forms created by the user each with separate "edit", "feedbacks", "delete" and "activate/deactivate" button.

b. *Create a new form*

Form creation is one of the primary goals of this project. A button named "Create New" is provided in the dashboard when the user wants to create a new form. After clicking this button one popup dialog opens and asks for the form title and a checkbox to set or unset anonymous data collection flag. This can be seen as above (Fig. 9).

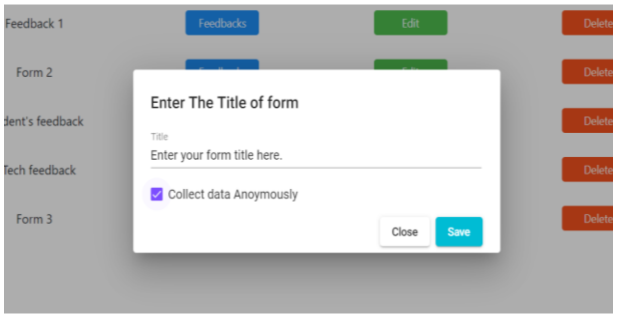
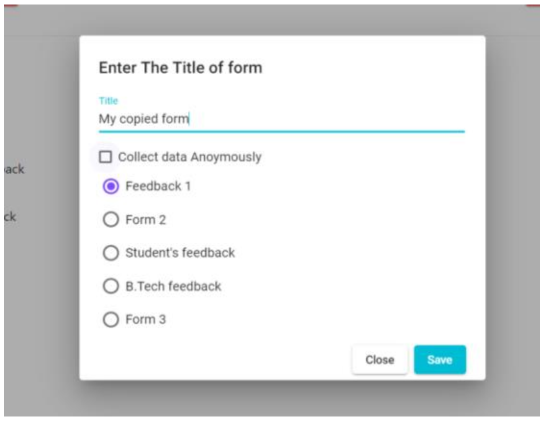


Fig 9: New Form creation

c. *Copy Existing Form*

The user also has the option to copy the existing form and modify it. The button is provided for this purpose. After clicking this button, a dialog pops up asking for form title, the form which is to be copied, and the checkbox to set or unset anonymous data collection flag as can be seen here (Fig. 10).


Enter The Title of form

Title: My copied form

☐ Collect data Anoymously

● Feedback 1

○ Form 2

○ Student's feedback

○ B.Tech feedback

○ Form 3

Fig 10: Copy an existing form

d. *Edit form*

The button is provided if the user wants to edit the existing form. This button click takes the user to Form Edit Page which is mentioned below in section 3 (Fig. 8).

e. *Delete form*

After the use of particular form user can have the option to delete it and its feedback database from the system.

f. *View Feedbacks*

To view feedbacks or responses obtained for a particular form, user can just click the “Feedbacks” button on the dashboard to view them and can also download the data collected. The details of this page named “Feedbacks Page” (Fig. 8) is explained below in section 6.

g. *Activate/ Deactivate form*

When the user wants to activate or deactivate a particular form, it can be done by using a slider provided in the row of the particular form as shown in Fig. 8.

3. Form Edit Page

The form edit page interface will open when a user had chosen to create a new form or edit an existing form. This page has various functionality for building the form from scratch by adding questions

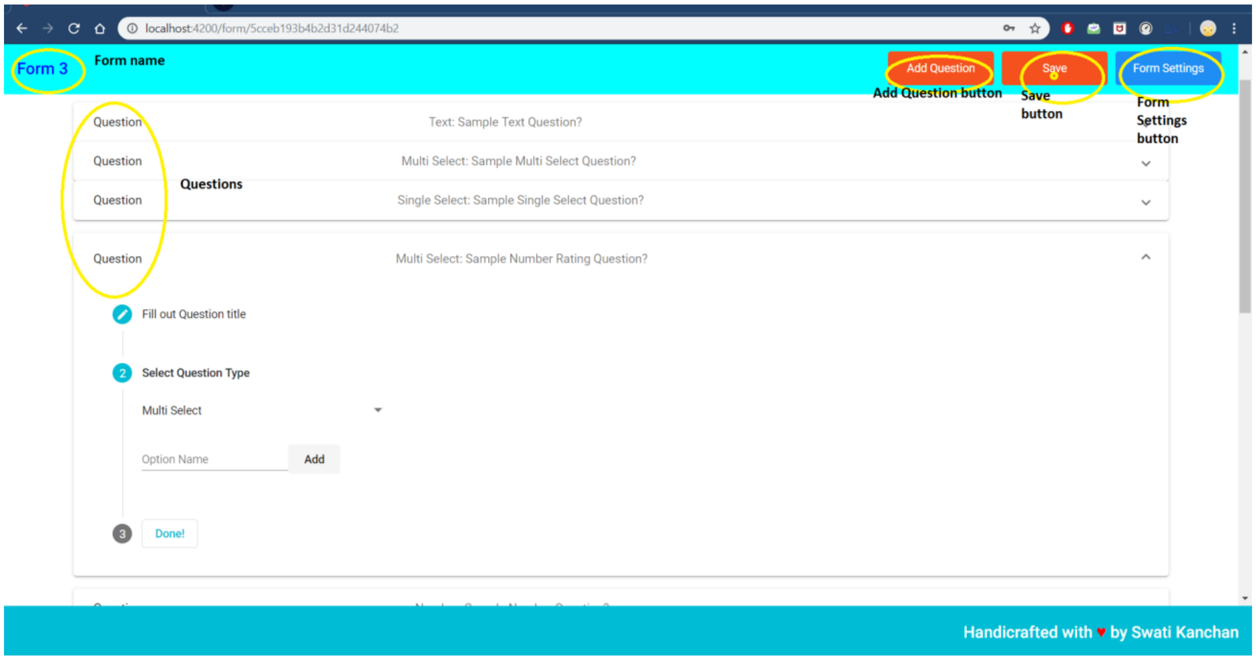

localhost:4200/form/5cceb193b4b2d31d244074b2
Form 3
Form name
Add Question
Save
Form Settings
Add Question button
Save button
Form Settings button
Question
Text: Sample Text Question?
Question
Multi Select: Sample Multi Select Question?
Questions
Question
Single Select: Sample Single Select Question?
Question
Multi Select: Sample Number Rating Question?
Fill out Question title
Select Question Type
Multi Select
Option Name
Add
Done!
Handicrafted with ♥ by Swati Kanchan

Fig 11: Form Edit page

of various types, setting iits settings, etc. The user interface of the form edit page is shown in this figure (Fig. 11).

a. Add questions

"Add Question" button is provided in the interface to add questions. After adding a question, the user has to follow the 3-step process as below:

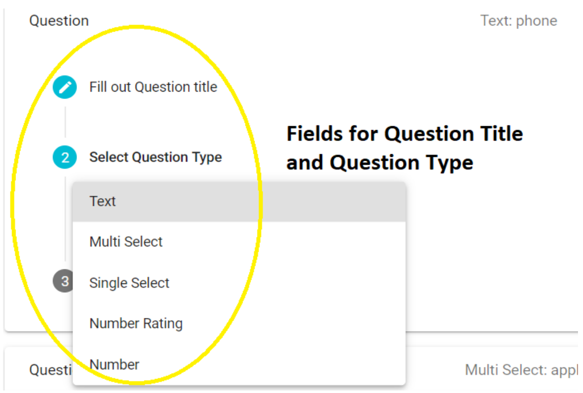


Fig 12: Adding questions

1. **Question title**
   At first, the user has to fill the title of the question to be added.
2. **Question type**
   User will also have to specify the type of question to be added.

3. **Possible answers**
   Depending upon the question type, the user will get the interface to add options or possible answers to the question.

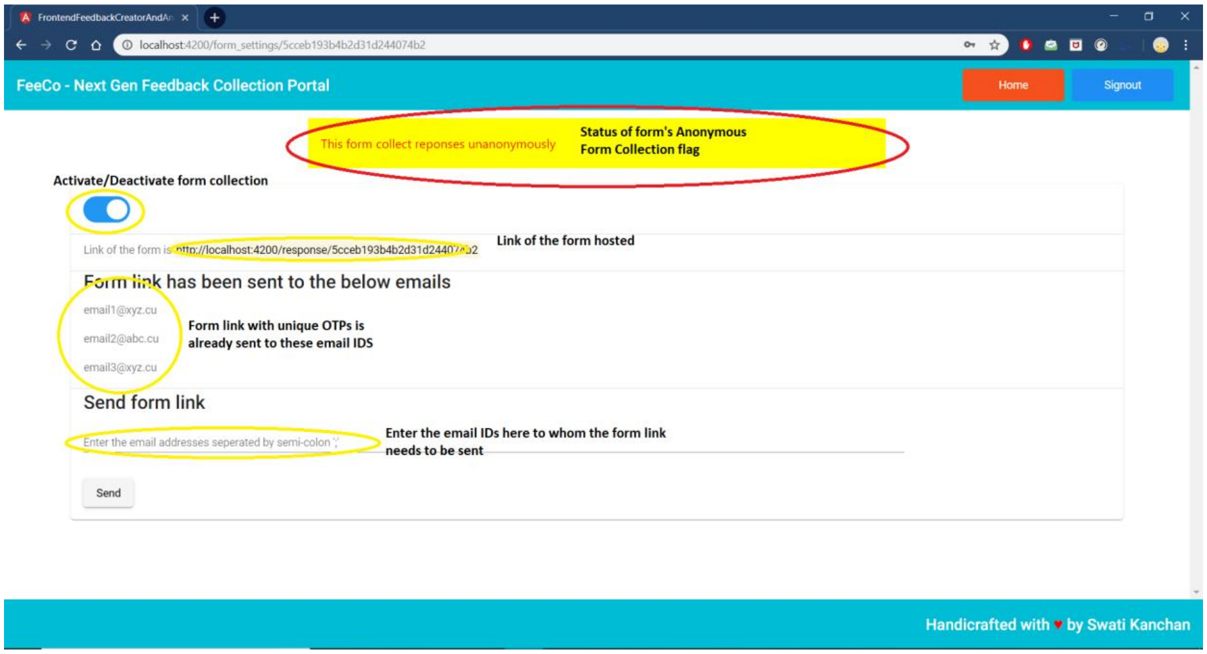
b. Save form

After the creation of all questions or editing them, the user must have to save the form to the system's database for further use. A button named “Save” is provided in the UI (Fig).

c. Form settings

“Form Settings” button present in the toolbar is used to set the functionality of the form and user level settings of the form. The detailed functions included in the settings are described below.

4. Form Settings Page



localhost:4200/form_settings/5cceb193b4b2d31d244074b2

FeeCo - Next Gen Feedback Collection Portal

This form collect reponses unanonymously

Status of form's Anonymous Form Collection flag

Activate/Deactivate form collection

Link of the form is http://localhost:4200/response/5cceb193b4b2d31d244074b2

Link of the form hosted

Form link has been sent to the below emails

email1@xyz.cu
email2@abc.cu
email3@xyz.cu

Form link with unique OTPs is already sent to these email IDS

Send form link

Enter the email addresses seperated by semi-colon ';'

Enter the email IDs here to whom the form link needs to be sent

Handicrafted with ♥ by Swati Kanchan

Fig 13: Form Settings page

Form settings page is the User Interface for user level settings of a particular form. Its basic functionalities are listed below. The UI of the Form settings is displayed in the image below (Fig. 13).

a. Activate/Deactivate button

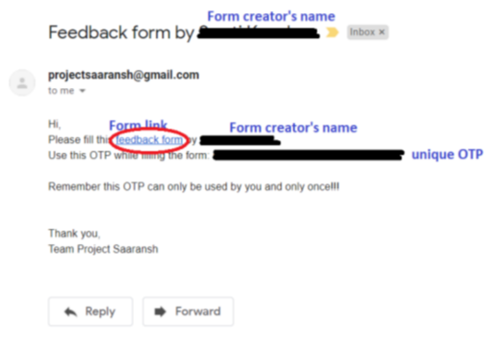
This button will be used to activate or deactivate the form submissions form the respondent.

b. Link of the form

Form submission link for the respondent

c. Add respondents by sending Emails with form link and unique OTP to respondents

This feature is used to add respondents for the submission of feedbacks. Respondents will receive an OTP which will be required while submitting the response or feedback. A sample email sent to one email ID is as shown alongside. (Fig. 14).



Feedback form by [redacted] Inbox ×

projectsaaransh@gmail.com
to me

Hi,
Please fill this feedback form by [redacted]
Use this OTP while filling the form: [redacted]

Remember this OTP can only be used by you and only once!!!

Thank you,
Team Project Saaransh

Fig 14: Email body with form link and OTP

d. View emails of the form link receivers

Emails of the already added respondents will be viewed in the UI

e. Status of form

Status of the form to collect responses anonymously or non-anonymously will be shown. This is to be submitted while creation of the form

f. View feedbacks

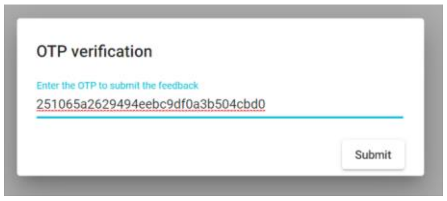
This will open a new page where respondents' responses can be viewed by the user for a particular form.

g. Summary of feedbacks

Short summary for all the responses collected for a particular question from a particular form can be viewed using this button.

5. Form link page

a. OTP verification dialog


OTP verification

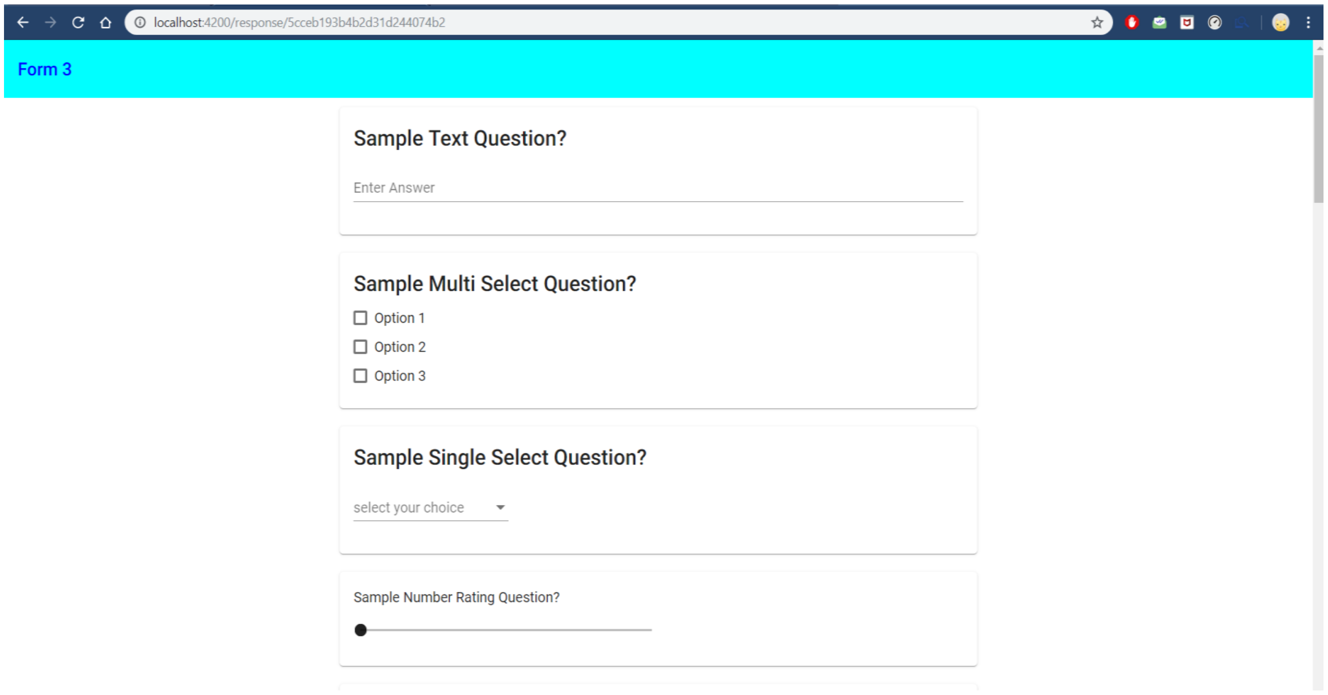
Enter the OTP to submit the feedback

251065a2629494eebc9df0a3b504cbd0

As soon as the respondent opens the link of the form, OTP verification dialog will be presented where the respondent will submit an OTP which they received via mail. After the verification of the OTP, the respondent can submit a response.

b. Questions and Answers fields

The form will contain the question set as per the particular form. A sample form when opened by a form-receiver through OTP is shown below in Fig. 15.


localhost:4200/response/5cceb193b4b2d31d244074b2

Form 3

Sample Text Question?

Enter Answer

Sample Multi Select Question?

☐ Option 1
☐ Option 2
☐ Option 3

Sample Single Select Question?

select your choice

Sample Number Rating Question?

Fig 15: Feedback form as viewed by the respondent

c. Submit responses

Respondent has to submit a response by typing in the input field provided. By clicking on the "Submit" Button, response or feedbacks will be submitted to the database of the application.

6. **Feedbacks Page**
   a. ***View respondents name, if form is not anonymous, with date and time***

      Feedbacks page will contain all the respondent name (if non-anonymous else the term “Respondent #”) along with submission time.

   b. ***Click on respondents to view responses***

      To view the response submitted by respondent user has to click on particular respondent.

   c. ***Export responses to CSV data***

      All responses from the respondents can be downloaded by user in the form of CSV file.

   d. ***Create a summary of “text comments” given by the respondent in feedbacks***

      The summary of the feedbacks using Machine learning algorithm will be provided in the UI.

   e. ***Export summary of the feedbacks to txt/pdf.***

      Summary of the feedbacks can be exported or downloaded by the user in the form of txt or pdf file.

➢ Backend and Database

NodeJS is an application runtime environment that allows us to write server-side applications in JavaScript. Thanks to its unique I/O model, it excels at the sort of scalable and real-time situations we are increasingly demanding of our servers. It’s also lightweight, efficient, and its ability to use JavaScript on both frontend and backend opens new avenues for development [*]. Therefore, we have used NodeJS for backend development of our application.

1. **Creation of Server**

   Node.js has a built-in module called HTTP, which allows Node.js to transfer data over the HyperText Transfer Protocol (HTTP). The

HTTP module can create an HTTP server that listens to server ports and gives a response back to the client.

The code for server creation is as follows:

```
//importing modules
const express = require('express');
const cors = require('cors');
const mongoose = require('mongoose');
mongoose.Promise = require('bluebird');
//mongoose.Promise = global.Promise;
const bodyparser = require('body-parser');
const app = express();
```

2. **Connection of server to MongoDB server**

   In order to connect to MongoDB, we need a URI string. A URI (Uniform Resource Identifier) is similar to a URL, and is supplied as a parameter to the mongo shell, Compass, and the MongoDB drivers when connecting to a MongoDB deployment. The URI string used helps in set up of authentication for our MongoDB instance, and a username and password is created for read and write access to a MongoDB database.

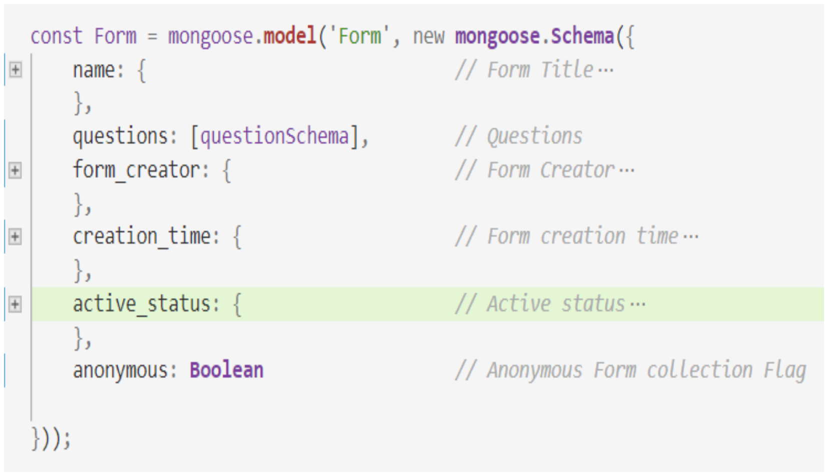
   The code for MongoDb server creation is as follows:

```
//connect to mongoDB
mongoose.connect('mongodb://localhost:27017/projectSaranshDB',
             {server: { poolSize: 5 }});
//on connection
 mongoose.connection.on('connected', ()=>{
   console.log("connected to database mongodb");
 });
 mongoose.connection.on('error', (err)=>{
   if(err){
       console.log("error in connection to database
mongodb.\n"+err);
   }
 });
```

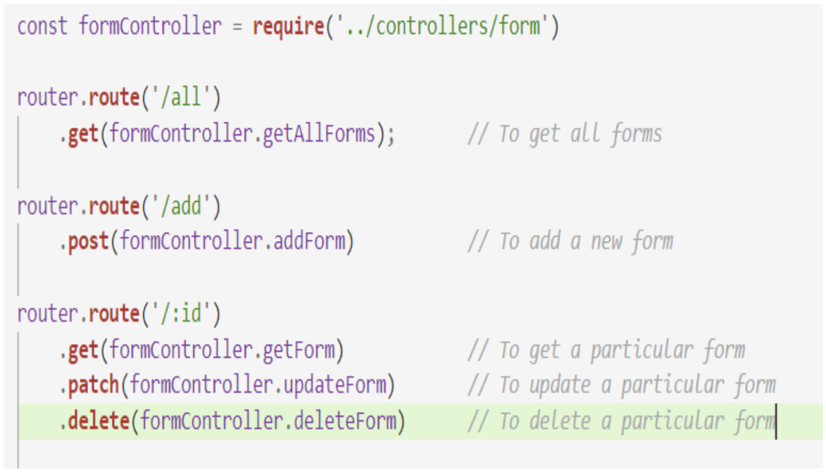
3. **Creation of Controllers, Models, and Routes for the following important modules:**

   A Controller of each of these following modules contain the main functions or methods required to carry important tasks related to that module. A Model contain the schema of the document related to the module which gets stored in MongoDB whereas the Routes contain the ExpressJS routes or the GET/POST APIs which are called by the application’s frontend to implement a specific feature. The model schema, and the important and basic routes along with the respective controller functions it is calling, for each of the modules are as shown below:

a. Form

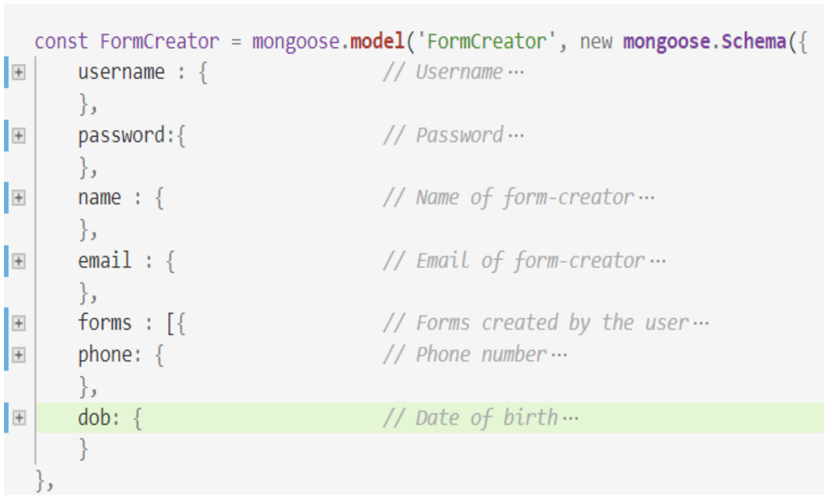


```
const Form = mongoose.model('Form', new mongoose.Schema({
    name: {                              // Form Title…
    },
    questions: [questionSchema],         // Questions
    form_creator: {                      // Form Creator…
    },
    creation_time: {                     // Form creation time…
    },
    active_status: {                     // Active status…
    },
    anonymous: Boolean                   // Anonymous Form collection Flag

}));
```

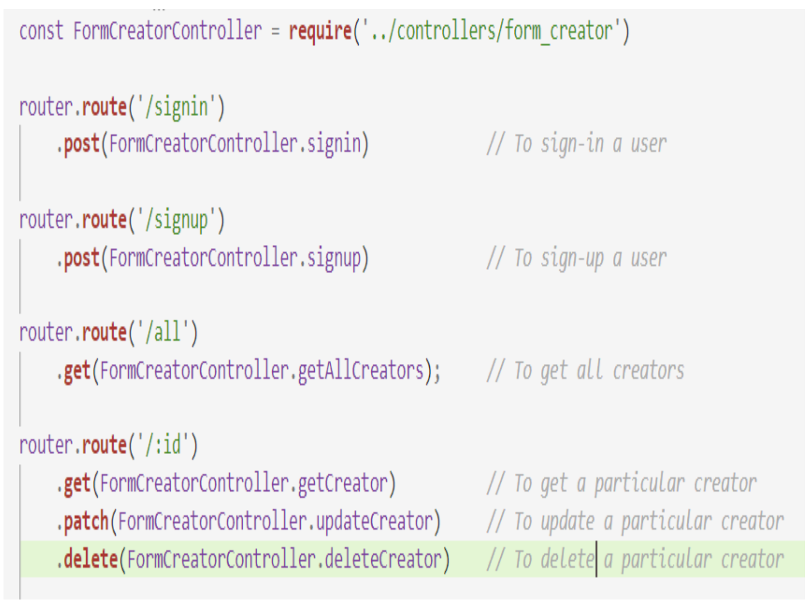


```
const formController = require('../controllers/form')

router.route('/all')
    .get(formController.getAllForms);       // To get all forms

router.route('/add')
    .post(formController.addForm)           // To add a new form

router.route('/:id')
    .get(formController.getForm)            // To get a particular form
    .patch(formController.updateForm)       // To update a particular form
    .delete(formController.deleteForm)      // To delete a particular form
```

b. Form creator

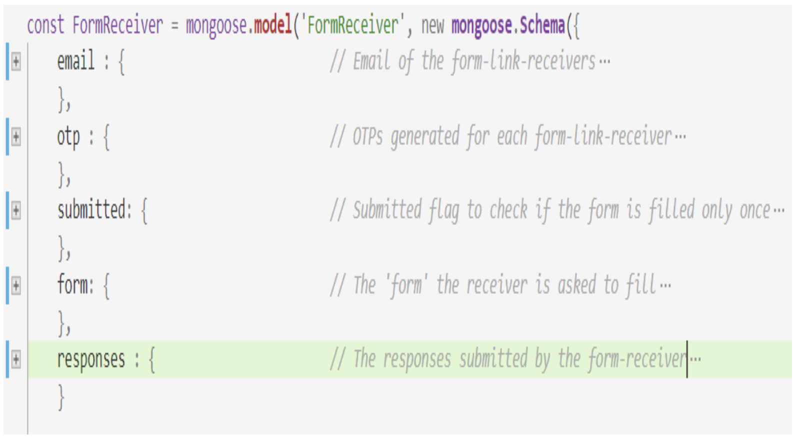


```
const FormCreator = mongoose.model('FormCreator', new mongoose.Schema({
    username : {                  // Username…
    },
    password:{                    // Password…
    },
    name : {                      // Name of form-creator…
    },
    email : {                     // Email of form-creator…
    },
    forms : [{                    // Forms created by the user…
    phone: {                      // Phone number…
    },
    dob: {                        // Date of birth…
    }
},
```

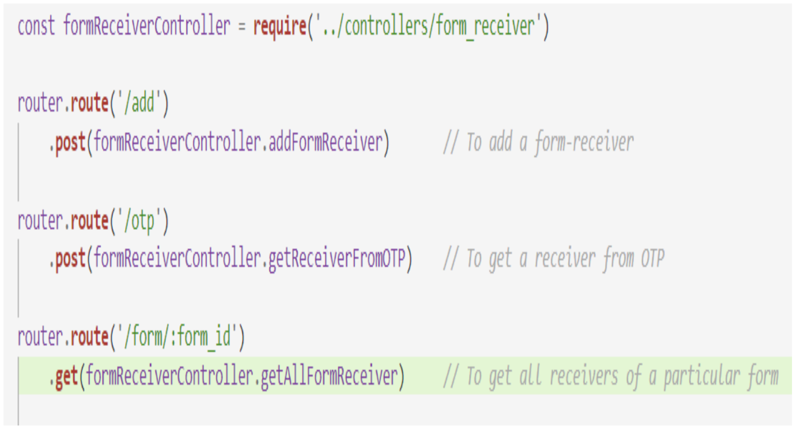


```
const FormCreatorController = require('../controllers/form_creator')

router.route('/signin')
    .post(FormCreatorController.signin)              // To sign-in a user

router.route('/signup')
    .post(FormCreatorController.signup)              // To sign-up a user

router.route('/all')
    .get(FormCreatorController.getAllCreators);      // To get all creators

router.route('/:id')
    .get(FormCreatorController.getCreator)           // To get a particular creator
    .patch(FormCreatorController.updateCreator)      // To update a particular creator
    .delete(FormCreatorController.deleteCreator)     // To delete a particular creator
```

c. Form receiver

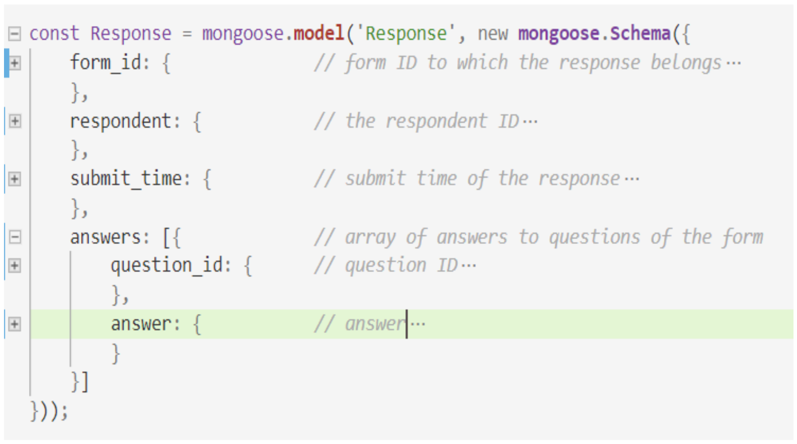


```
const FormReceiver = mongoose.model('FormReceiver', new mongoose.Schema({
    email : {                       // Email of the form-link-receivers···
    },
    otp : {                         // OTPs generated for each form-link-receiver···
    },
    submitted: {                    // Submitted flag to check if the form is filled only once···
    },
    form: {                         // The 'form' the receiver is asked to fill···
    },
    responses : {                   // The responses submitted by the form-receiver···
    }
```

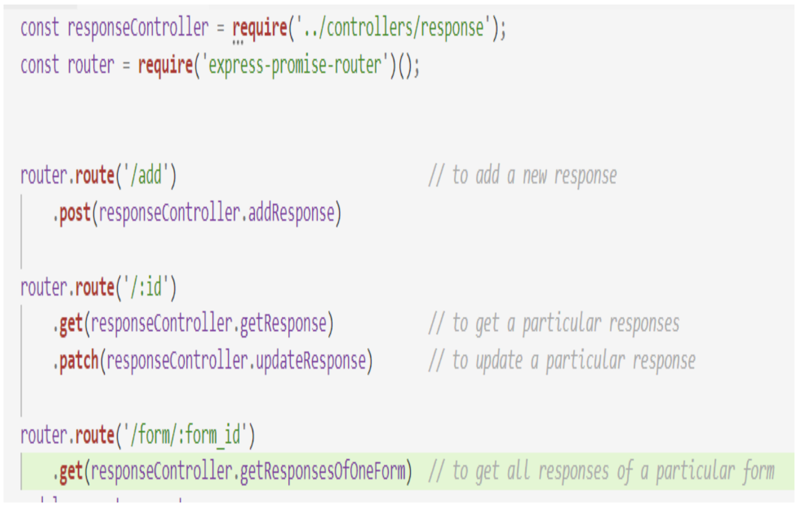


```
const formReceiverController = require('../controllers/form_receiver')

router.route('/add')
    .post(formReceiverController.addFormReceiver)        // To add a form-receiver

router.route('/otp')
    .post(formReceiverController.getReceiverFromOTP)     // To get a receiver from OTP

router.route('/form/:form_id')
    .get(formReceiverController.getAllFormReceiver)      // To get all receivers of a particular form
```

d. Response



```
const Response = mongoose.model('Response', new mongoose.Schema({
    form_id: {              // form ID to which the response belongs…
    },
    respondent: {           // the respondent ID…
    },
    submit_time: {          // submit time of the response…
    },
    answers: [{             // array of answers to questions of the form
        question_id: {      // question ID…
        },
        answer: {           // answer…
        }
    }]
}));
```



```
const responseController = require('../controllers/response');
const router = require('express-promise-router')();


router.route('/add')                                    // to add a new response
    .post(responseController.addResponse)

router.route('/:id')
    .get(responseController.getResponse)                // to get a particular responses
    .patch(responseController.updateResponse)           // to update a particular response

router.route('/form/:form_id')
    .get(responseController.getResponsesOfOneForm)      // to get all responses of a particular form
```

4. Initiating MongoDB database connection

To create a database in MongoDB, we start by creating a MongoClient object, then specify a connection URL with the correct IP address and the name of the database we want to create. MongoDB will create the database if it does not exist, and make a connection to it. Below is our code:

```
const port = process.env.PORT || app.get('port') || 3000;

app.listen(port, () => {
 console.log("server started at port: " + port);
});
```

➢ Text Summarization

1. Machine Learning Model

Machine learning (ML) is a category of algorithm that allows software applications to become more accurate in predicting outcomes without being explicitly programmed. We are using the pre-trained Machine Learning model, available in the Python library Sumy, and storing it in the same cloud where our website is hosted. It summarizes the Feedback texts received from every student automatically using the pre-trained weights.

Extractive text summarization techniques perform summarization by picking portions of texts and constructing a summary, unlike abstractive techniques which conceptualize a summary and paraphrases it. There are various techniques under abstractive summarization each technique is implemented differently based on researchers approaches, these techniques include clustering, graph theory, lexical chains, word-net etc., some are statistical in nature others deep rooted in linguistics while others robustly try to combine two or more techniques.

Sumy: Simple library and command line utility for extracting summary from HTML pages or plain texts. The package also contains simple evaluation framework for text summaries [9]. It contains various implementations such as Luhn, Text Rank, Lex Rank, etc. The summarization technique we have used is Latent Semantic Analysis.

2. Connecting Text summarization model with Backend

We have called the python library Sumy from our NodeJS backend by creating a child process and then calling a python file.

```
const spawn = require("child_process").spawn;
const pythonProcess = spawn('python',["python.py"]);
pythonProcess.stdout.on('data', (data) => {
 console.log(" That was an easy " + data);
});
```

3. Predicting Summarization data from responses

Latent Semantic Analysis [12] is an unsupervised method of summarization it combines term frequency techniques with singular value decomposition to summarize texts. It is one of the most recent suggested technique for summarization. LSA (Latent Semantic Analysis) also known as LSI (Latent Semantic Index) LSA uses bag of word (BoW) model, which results in a term-document matrix (occurrence of terms in a document). Rows represent terms and columns represent documents. LSA learns latent topics by performing a matrix decomposition on the document-term matrix using Singular value decomposition. LSA is typically used as a dimension reduction or noise reducing technique.

Python code:

```
from sumy.summarizers.lsa
import LsaSummarizer
summarizer_2 = LsaSummarizer()
summary_2 = summarizer_2(parser.document, 2)
for sentence in summary_2:
  print(sentence)
```

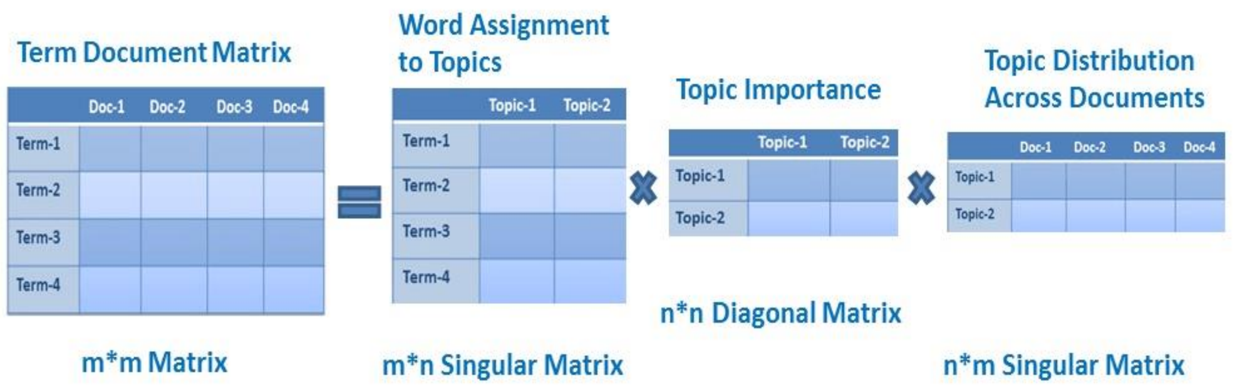


Fig 16: Term document matrix

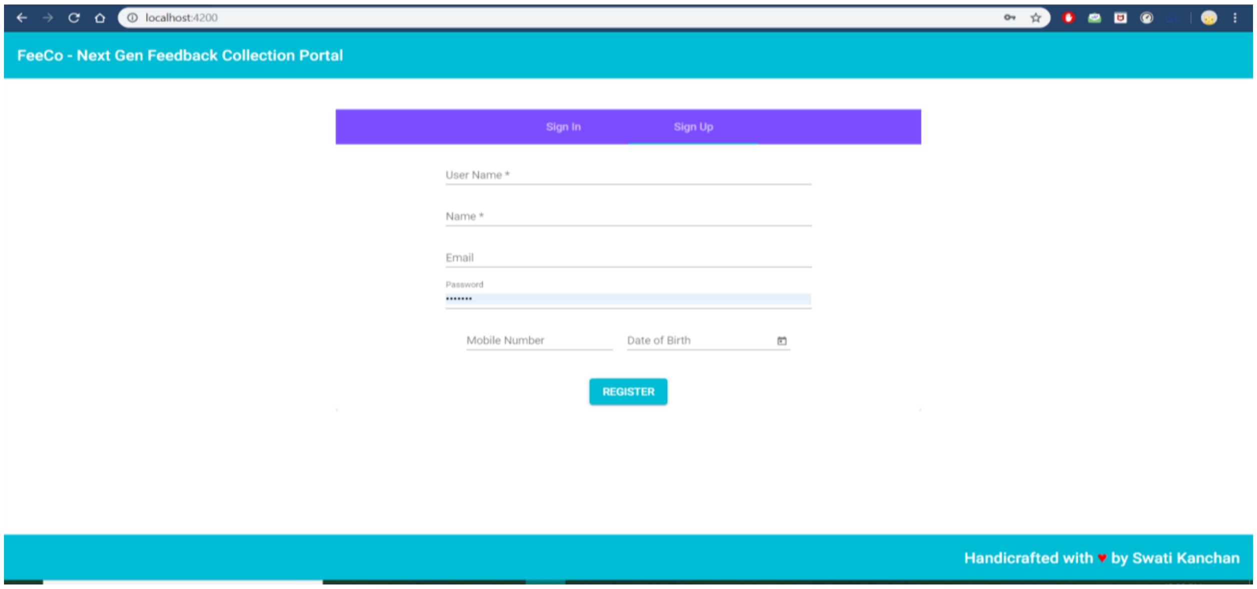
4. Exporting Summary to Txt/Pdf file

After using the machine learning model for summarization of the feedbacks or responses, the results are stored in database. This summary can be downloaded by the user in the form of txt or PDF file.

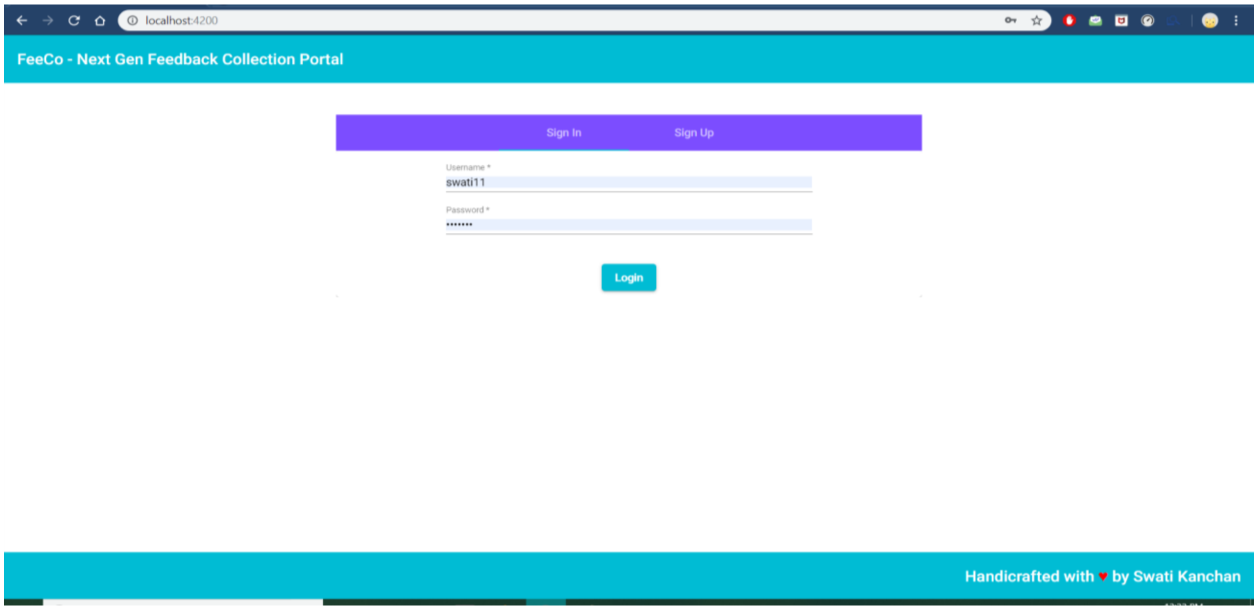
Demonstration

Following is a basic demonstration of how this tool work from the beginning. This have been explained using easy steps along with the screenshots of application in every stage.

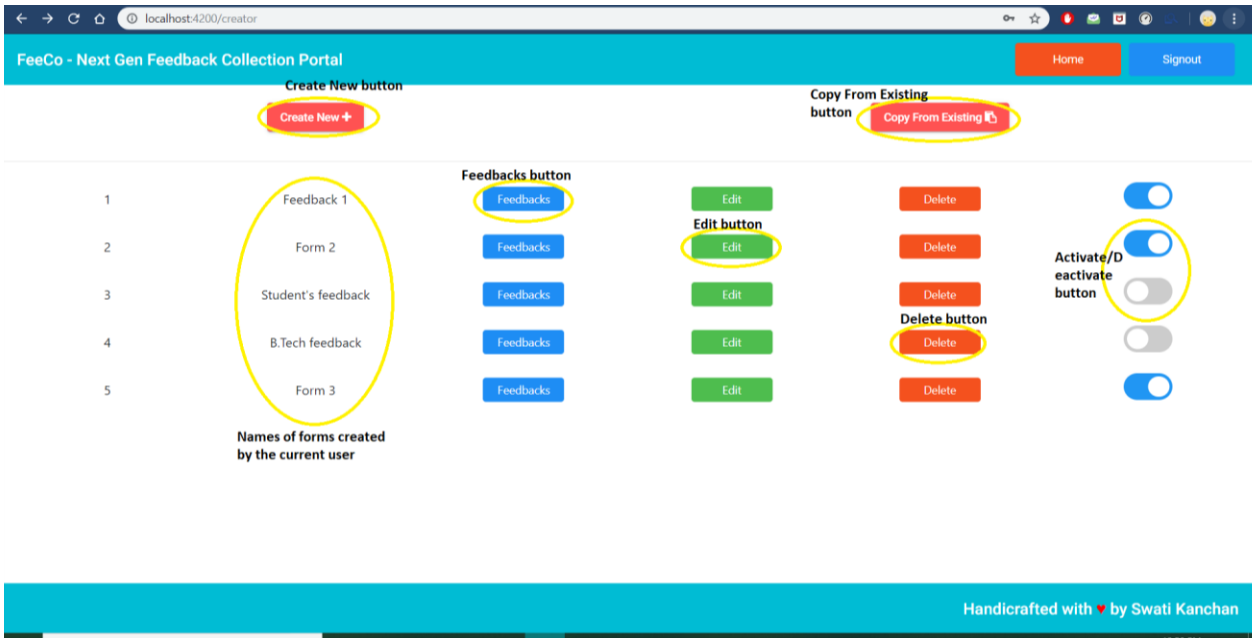
Step 1: Register as a new user with the following details.



Step 2: Sign-up with the username and password you had set while registering.

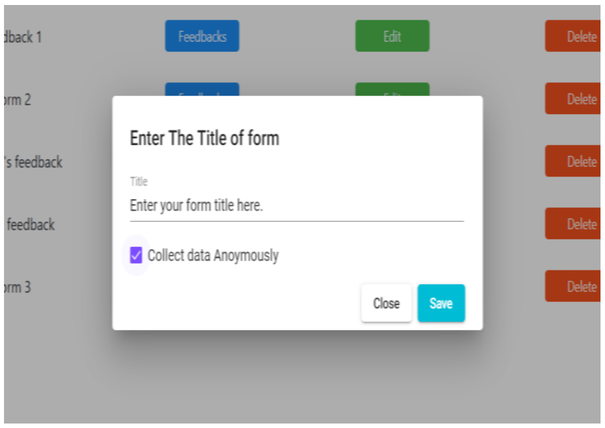

localhost:4200
FeeCo - Next Gen Feedback Collection Portal
Sign In
Sign Up
Username *
swati11
Password *
Login
Handicrafted with ♥ by Swati Kanchan

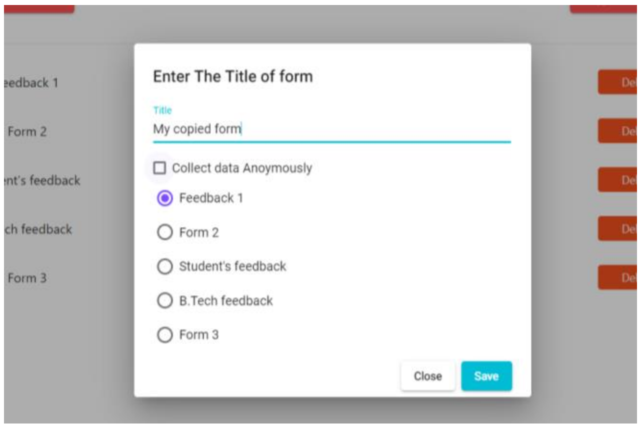
Step 3: This is the user Dashboard to view, edit, activate/deactivate or delete a form.


localhost:4200/creator
FeeCo - Next Gen Feedback Collection Portal
Home
Signout
Create New button
Create New +
Copy From Existing button
Copy From Existing
Feedbacks button
Edit button
Delete button
Activate/Deactivate button
Names of forms created by the current user
Handicrafted with ♥ by Swati Kanchan

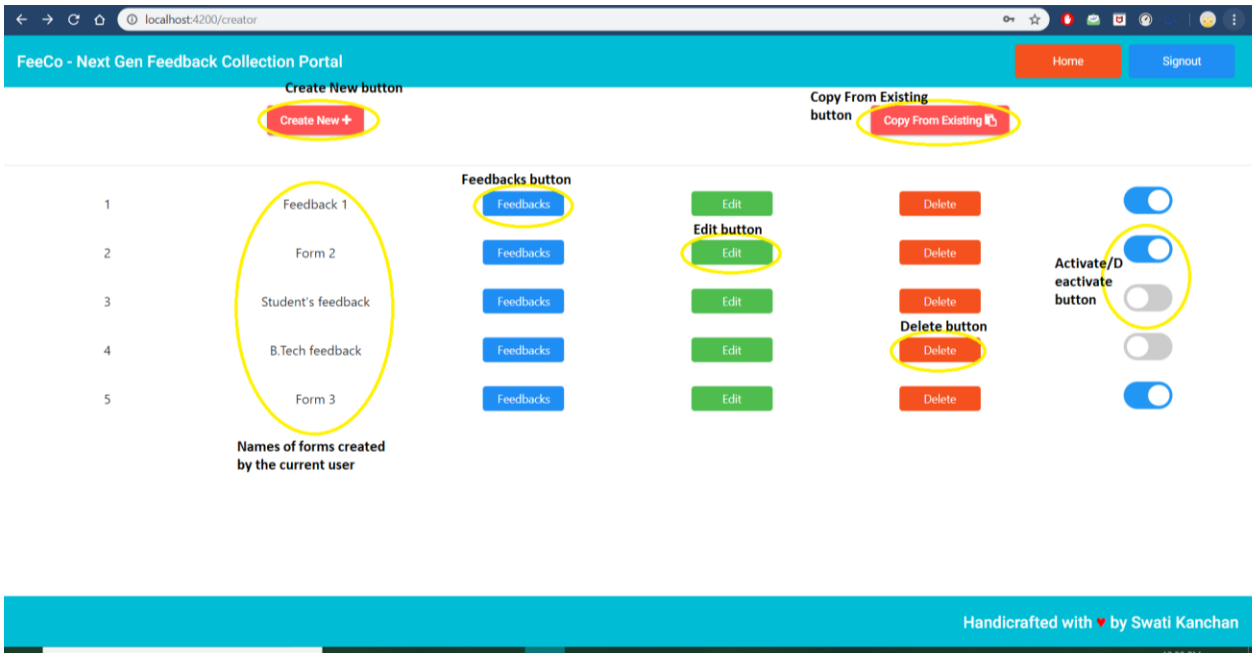
1	Feedback 1	Feedbacks	Edit	Delete	●
2	Form 2	Feedbacks	Edit	Delete	●
3	Student's feedback	Feedbacks	Edit	Delete	○
4	B.Tech feedback	Feedbacks	Edit	Delete	○
5	Form 3	Feedbacks	Edit	Delete	●

Step 4: User can create new forms either a) by clicking on "Create New" button on top left to create a new form from scratch or b) by clicking on "Copy From Existing" button on top right to copy from an existing form and then edit it.



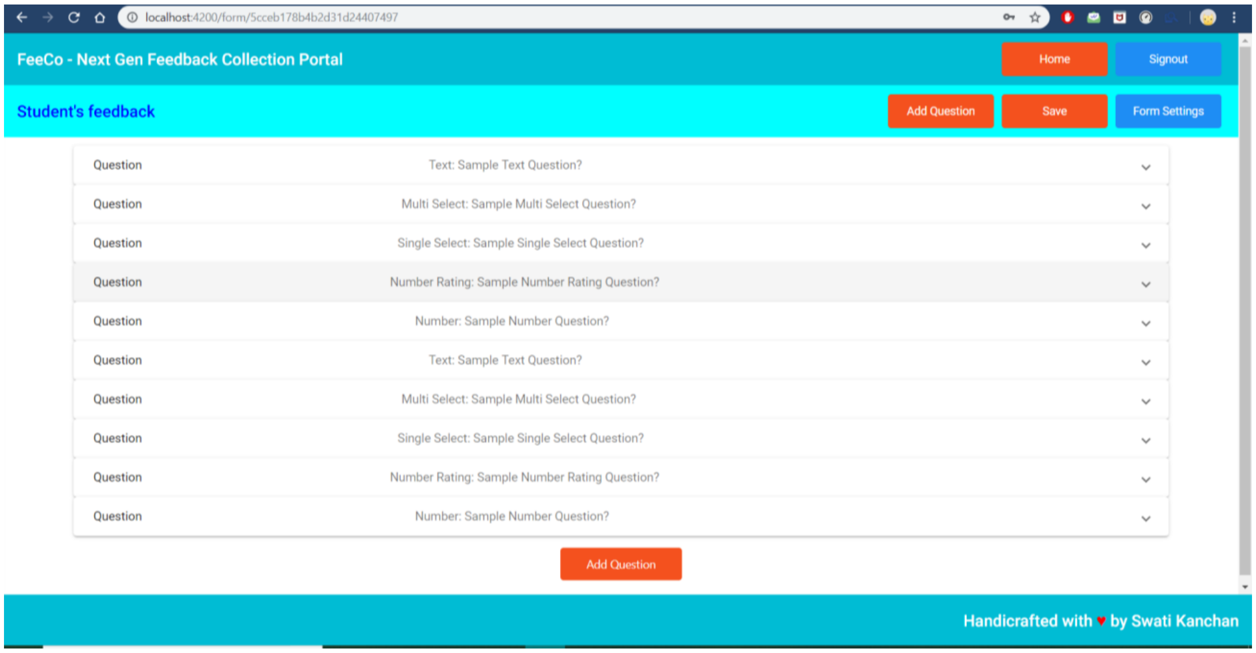


Step 5: Click on "Edit" button to edit it.


localhost:4200/creator
FeeCo - Next Gen Feedback Collection Portal
Home
Signout
Create New button
Create New +
Copy From Existing button
Copy From Existing
Feedbacks button
Edit button
Delete button
Activate/Deactivate button
Names of forms created by the current user
Handicrafted with ♥ by Swati Kanchan

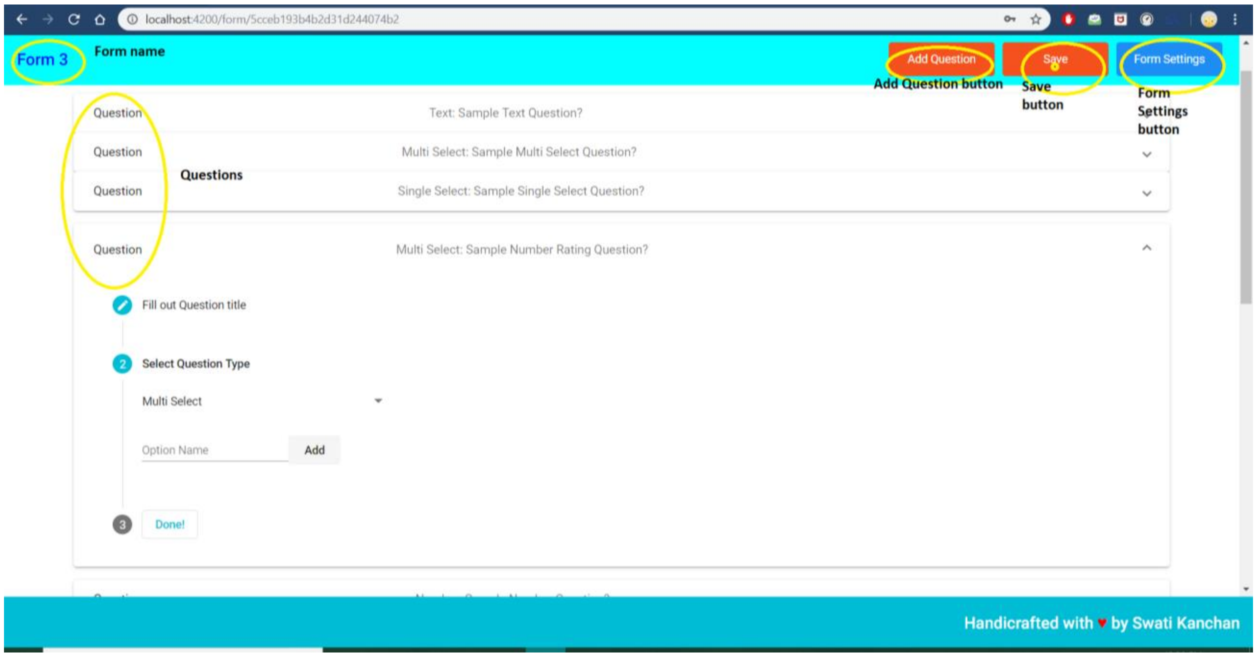
1	Feedback 1	Feedbacks	Edit	Delete	
2	Form 2	Feedbacks	Edit	Delete	
3	Student's feedback	Feedbacks	Edit	Delete	
4	B.Tech feedback	Feedbacks	Edit	Delete	
5	Form 3	Feedbacks	Edit	Delete	

Step 6: The questions are shown in the edit-form page. Five different sample questions are already present in a new form. Click on one of it to edit a question and click on "Add Question" to add more.


localhost:4200/form/5cceb178b4b2d31d24407497
FeeCo - Next Gen Feedback Collection Portal
Home
Signout
Student's feedback
Add Question
Save
Form Settings
Add Question
Handicrafted with ♥ by Swati Kanchan

Question	Text: Sample Text Question?
Question	Multi Select: Sample Multi Select Question?
Question	Single Select: Sample Single Select Question?
Question	Number Rating: Sample Number Rating Question?
Question	Number: Sample Number Question?
Question	Text: Sample Text Question?
Question	Multi Select: Sample Multi Select Question?
Question	Single Select: Sample Single Select Question?
Question	Number Rating: Sample Number Rating Question?
Question	Number: Sample Number Question?

Step 7: Add question Title, Type, and possible answers (only in case of multiple and single choice questions) and hit done. Finally Click on "Save" button to save the changes. To make more changes click on "Form Settings" button.

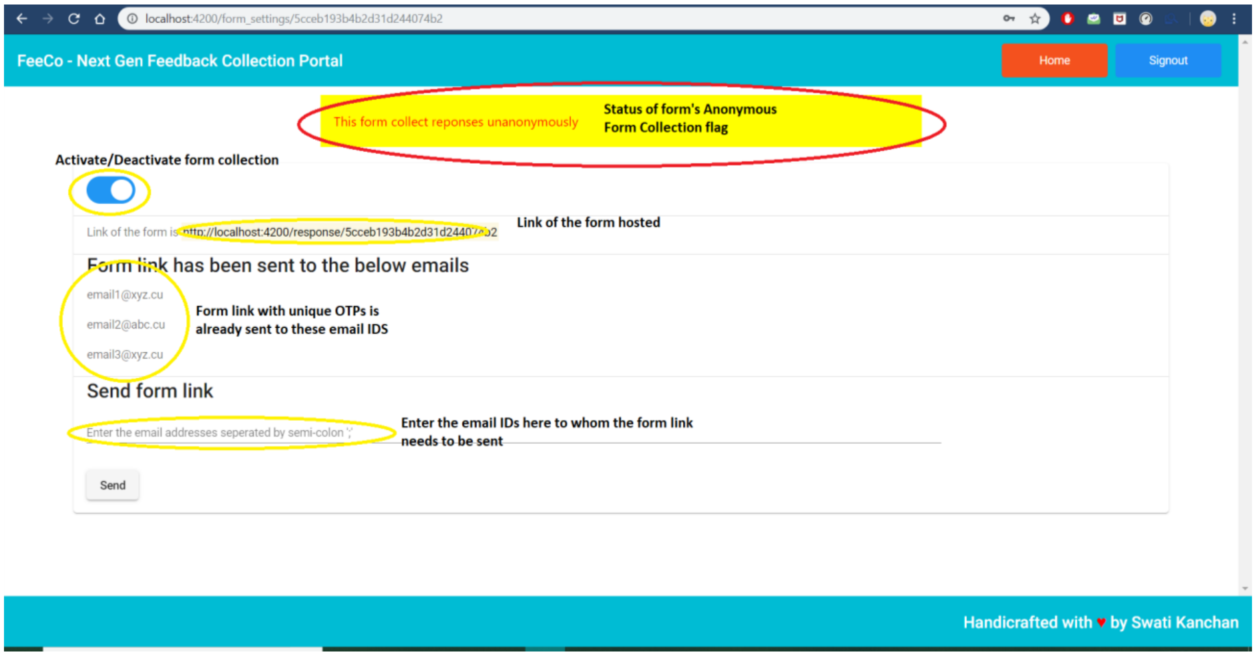

localhost:4200/form/5cceb193b4b2d31d244074b2

Question	Text: Sample Text Question?
Question	Multi Select: Sample Multi Select Question?
Question	Single Select: Sample Single Select Question?
Question	Multi Select: Sample Number Rating Question?

1. Fill out Question title
2. Select Question Type
   Multi Select
   Option Name — Add
3. Done!

Handicrafted with ♥ by Swati Kanchan

Step 8: Enable/Disable the form, get the link to current form, send emails with form links. Also see if the form collects responses anonymously or not.


localhost:4200/form_settings/5cceb193b4b2d31d244074b2

FeeCo - Next Gen Feedback Collection Portal

This form collect reponses unanonymously

Status of form's Anonymous Form Collection flag
Activate/Deactivate form collection
Link of the form hosted

Link of the form is http://localhost:4200/response/5cceb193b4b2d31d244074b2

Form link has been sent to the below emails

email1@xyz.cu
email2@abc.cu
email3@xyz.cu

Form link with unique OTPs is already sent to these email IDS

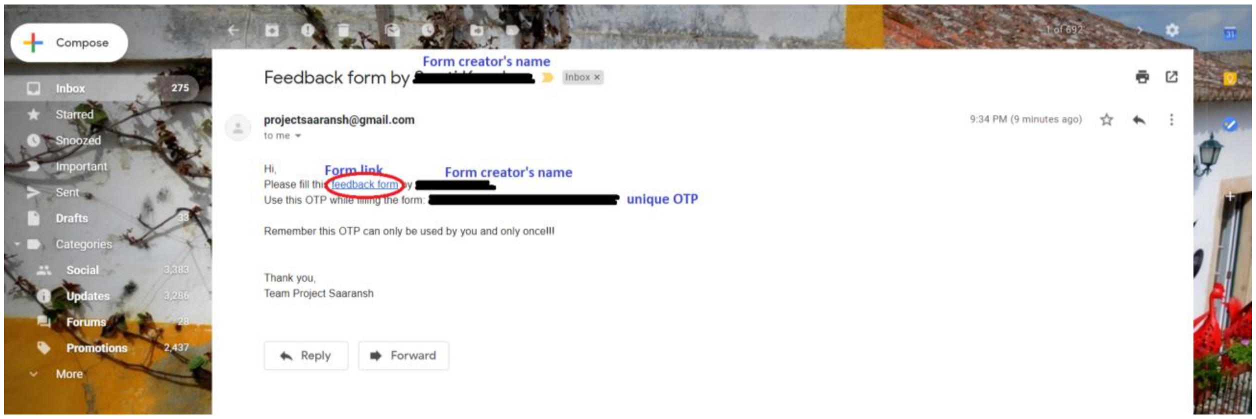
Send form link

Enter the email addresses seperated by semi-colon ';'

Enter the email IDs here to whom the form link needs to be sent
Send

Handicrafted with ♥ by Swati Kanchan

Step 9: Emails are received in this way, with a form link, form-creator's name and a unique OTP. Click on the form link.



Feedback form by [redacted]

projectsaaransh@gmail.com
to me

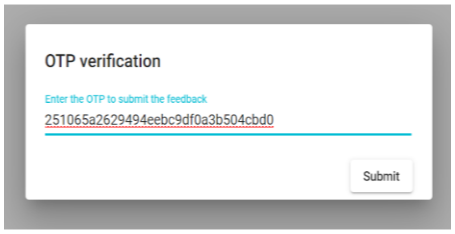
9:34 PM (9 minutes ago)

Hi,
Please fill this feedback form by [redacted]
Use this OTP while filling the form: [redacted]

Remember this OTP can only be used by you and only once!!!

Thank you,
Team Project Saaransh

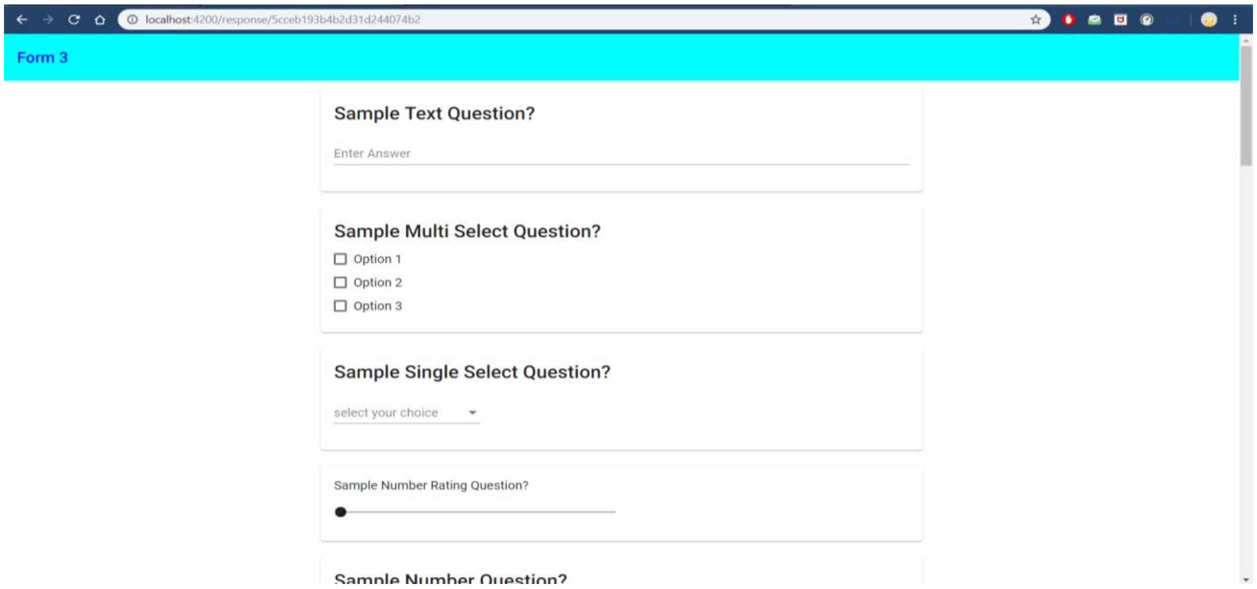
Step 10: Enter the OTP sent in the mail to get access of the form and fill.



OTP verification

Enter the OTP to submit the feedback
251065a2629494eebc9df0a3b504cbd0

Step 11: Fill the form.



localhost:4200/response/5cceb193b4b2d31d244074b2

Form 3

Sample Text Question?
Enter Answer

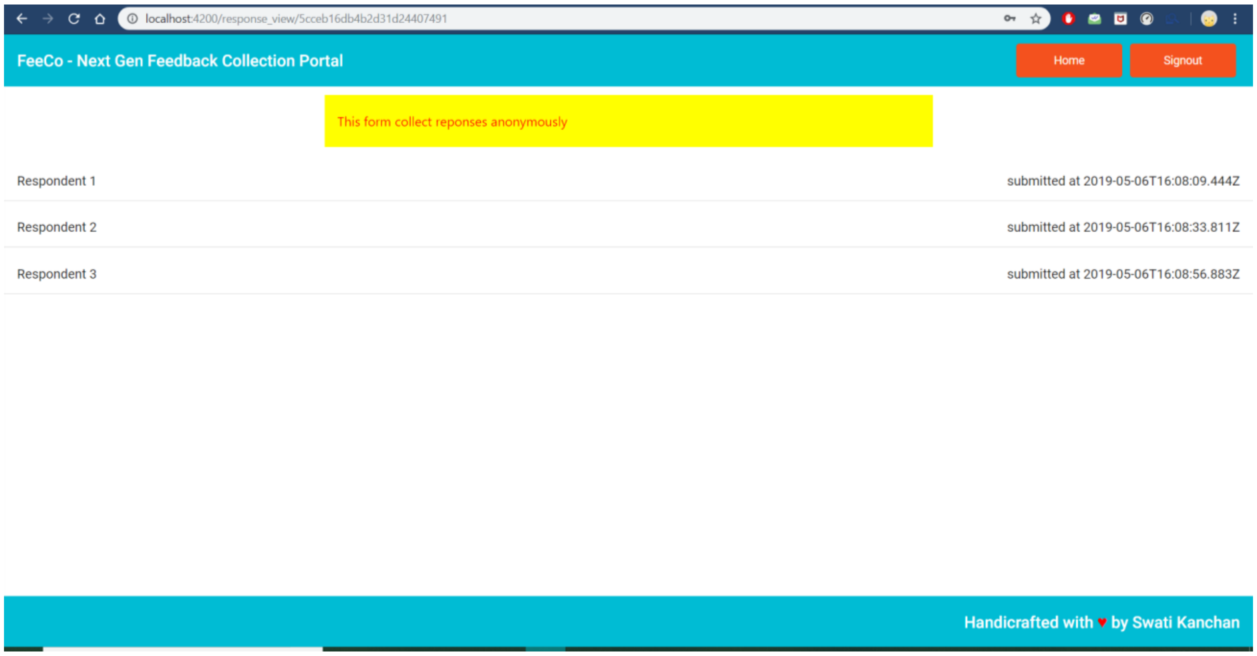
Sample Multi Select Question?
☐ Option 1
☐ Option 2
☐ Option 3

Sample Single Select Question?
select your choice

Sample Number Rating Question?

Sample Number Question?

Step 12: The form-creator can view the responses after clicking on the "Feedbacks" button of the form on the Dashboard. Based on whether the form takes responses anonymously or non-anonymously, respondents' email IDs or just the term "Respondent #" are shown respectively. Click on a respondent to view their response.


localhost:4200/response_view/5cceb16db4b2d31d24407491

FeeCo - Next Gen Feedback Collection Portal

Home Signout

This form collect reponses anonymously

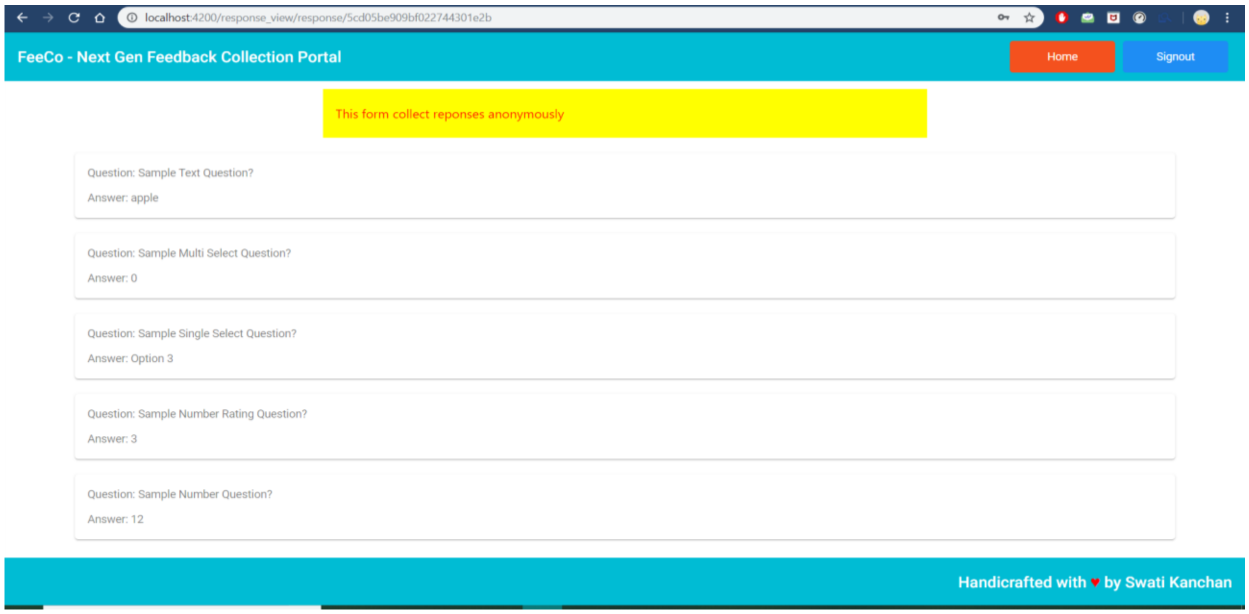
Respondent 1 submitted at 2019-05-06T16:08:09.444Z

Respondent 2 submitted at 2019-05-06T16:08:33.811Z

Respondent 3 submitted at 2019-05-06T16:08:56.883Z

Handicrafted with ♥ by Swati Kanchan

Step 13: View the response of the respondent. Each question is mapped with its answer given.


localhost:4200/response_view/response/5cd05be909bf022744301e2b

FeeCo - Next Gen Feedback Collection Portal

Home Signout

This form collect reponses anonymously

Question: Sample Text Question?
Answer: apple

Question: Sample Multi Select Question?
Answer: 0

Question: Sample Single Select Question?
Answer: Option 3

Question: Sample Number Rating Question?
Answer: 3

Question: Sample Number Question?
Answer: 12

Handicrafted with ♥ by Swati Kanchan

Conclusions and Future Work

In this project, we addressed the problem of lack of open source tool for collection of feedbacks and its analysis. One of the main contributions of our work is to create an interface for a user to create a form and send this form to the respondent without much effort. This interface currently has many bests in class features such as sending emails to the respondent without opening other third-party apps, Collection of data anonymously or non-anonymously, verifying user’s identity with OTP, feedbacks analysis and summary using Machine learning Methods, downloading feedback data, etc.

The main focus of our thesis was on the creation of the forms and sending forms to the user. We have also introduced machine learning based Latent semantic analysis model for summarization and analysis of the feedbacks and responses from the respondent. A new approach based on MEAN stack was used for this project. This project will help various sectors such as Education, Advertisement, business, etc., to excel in their fields by collecting valuable feedbacks and suggestions and analyzing them further in this single platform.

Many diﬀerent adaptations, tests, and experiments have been left for the future due to lack of time (i.e. the experiments with real data are usually very time consuming, requiring even days to finish a single run). Future work concerns deeper analysis of particular mechanisms, new proposals to try diﬀerent methods, or simply curiosity.

There are some ideas that can improve our existing next generation feedback collection platforms. Below are some our future works:

- Multilingual forms: Support of many regional and international languages for the respondent.
- Question types: Currently our form supports 5 types of question, we can add more question types as per the requirement.
- Designing custom feedback survey appearance: User can add various design and look to the feedback form page for the respondent as per the requirement.
- Offline functionality: Creating a desktop app or chrome app for the users who use this application frequently.

- Visualize data response: Create statistics and engaging graphs with a few clicks for all questions of our feedback form. Filter the results based on certain responses. Export the statistics to Excel or PDF.
- Dynamic questions: Create questions based on previous answers. Use answers given to your previous questions to validate and refine your results.

References

1. "What is survey?" , Quadratics, https://www.qualtrics.com/experience-management/research/survey-basics/ Accessed on:
2. Survey Monkey, **https://www.surveymonkey.com/**, Accessed on Mar 2019
3. Lime Survey, **https://www.limesurvey.org/first-start**, Accessed on Mar 2019
4. NodeJS, **https://nodejs.org/**, Accessed on Aug 2018
5. ExpressJS, **https://expressjs.com/**, Accessed on Oct 2018
6. Angular, **https://angular.io/**, Accessed on Oct 2018
7. MongoDB, **https://www.mongodb.com/**, Accessed on Oct 2018
8. MongooseJS, **https://mongoosejs.com/**, Accessed on Oct 2018
9. Github **https://github.com/miso-belica/sumy**, Accessed on Nov 2018
10. Python, **https://www.python.org/**, Accessed on Oct 2018
11. Eric Ondenyi, "Extractive Text Summarization Techniques With sumy", Available on **https://medium.com/@ondenyi.eric/extractive-text-summarization-techniques-with-sumy-3d3b127a0a32**, Accessed on Nov 2018
12. Avinash Navlani, "Latent Semantic Analysis using Python", Availble on **https://www.datacamp.com/community/tutorials/discovering-hidden-topics-python**, Accessed on Jan 2019
13. NeverForgetY2K, "How to call a Python function from Node.js", Available on **https://stackoverflow.com/questions/23450534/how-to-call-a-python-function-from-node-js**, Accessed on Feb 2019