Online Examination Portal with Proctoring

Submitted By

Setti Naveen Naidu	- 191ME177
P Gagan Kumar Chowdary	- 191ME160
G Tejesh	- 191ME288

Under the guidance of,

Dr. Sowmya Kamath S

Dept of IT , NITK Surathkal

Ms. Thanmayee S

Dept of IT , NITK Surathkal

In partial fulfillment for the award of the degree of

Bachelor of Technology

in

Information Technology

at

Department of Information Technology

National Institute of Technology Karnataka, Surathkal

Abstract

With the current disruption that happened in the education sector because of COVID-19, one question that arises in mind of everybody is about examinations that have been delayed and uncertainty that has made the students worried about their academic career. As a result we created an online examination portal for safe and secure examinations . The main objective of this Online Examination Portal is to efficiently evaluate the candidate thoroughly through a fully automated system that not only saves lot of time but also gives fast results. This can be used in educational institutions . It can be used anywhere any time as it is a web based application (user location doesn't matter). No restriction that examiner has to be present when the candidate.

Contents

CHAPTER -1 . Introduction

- Education is undergoing a huge transformation with emerging technologies Universities and Colleges are considering how their examination schedules should proceed. They are exploring for options to conduct safe and secure online examinations
- Proctoring examinations is one part of the larger problem statement on how to evaluate students on their academic performance on an ongoing basis
- We came up with a web application which helps to conduct examinations in safe and secure online mode.
- Our Website helps to conduct examinations during this tough time, our website is built with live proctoring features with AI-assisted facial recognition system and raises an alert when Candidate may be violating the rules outlined in the test

CHAPTER - 2 . Requirements Analysis

➢ i)Functional Requirements :

- Online registration is required (students , teachers)
- Secured login is required
- AI assisted live proctoring is required
- Multiple variety of questions like mcqs should be framed
- Time restriction , alerts when user violates rules
- Review results is required

➢ ii)Non-Functional Requirements :

- Consistent and reliable
- Responsive Design – Have to work well for different types of screens
- Support different types of browsers
- User friendly
- Eco friendly
- Error message should be clear
- Less Graphics

➢ iii)Use case Scenarios :

❖ Student :

- Student click on sign up
- System asks for info
- Student click on login and provides accurate credentials
- Students attempts the exam with camera on
- System takes the input of answers
- Students ask for results
- System gives result

❖ Teacher :

- Teacher click on sign up
- System asks for info
- Teacher click on login and provides accurate credentials
- Teacher wants to create an exam
- System gives access to add or edit questions
- System records information given by teacher
- Teacher makes it access to students by providing credentials
- Teacher wants to view the results
- System gives the results

CHAPTER – 3 .System Design

➢ i)Design Goals :

- **Visual Appeal :**
  To maintain our web application look and feel good.

- **Navigability :**
  To maintain our web application easy to navigate by using navigation bars , social media icons and also less graphics to reduce load time.

- **Compatibility :**
  To make our web application appear functional on different web browsers.

- **Simplicity :**
  To make it accessible to as many as users possible by using simple words for buttons etc

➢ ii)Methodology :

We used Model View Control (MVC) design pattern. It is a pattern that divides the code into three parts .This is done to distinguish internal representations of data from how data is presented to and accepted by users.

The application was developed using different programming models and languages which include **HTML**, **CSS** and **JS** (for the front-end interface) which comes under view section and **NodeJs** & **ExpressJs** which comes under controller section, **Mongodb** (for the backend) which comes under Model section and served through a web server.

The use of HTML and CSS, which is a markup language for information presentation and a styling language respectively, allow for the user-interface to be designed and properly laid out. To enable dynamic content generation, Nodejs, ExpessJs is used to generate dynamic contents based on the user of the system and the corresponding content stored in the backend database which is managed by Mongodb. In other words, the computer simply acts as the medium for students to take examinations, for teachers to conduct tests, and for the transmission of test papers.

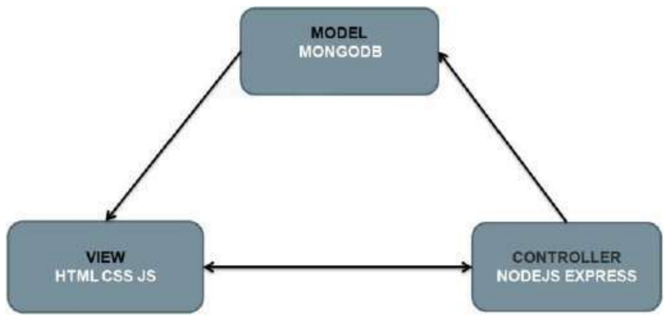


Fig 3.1

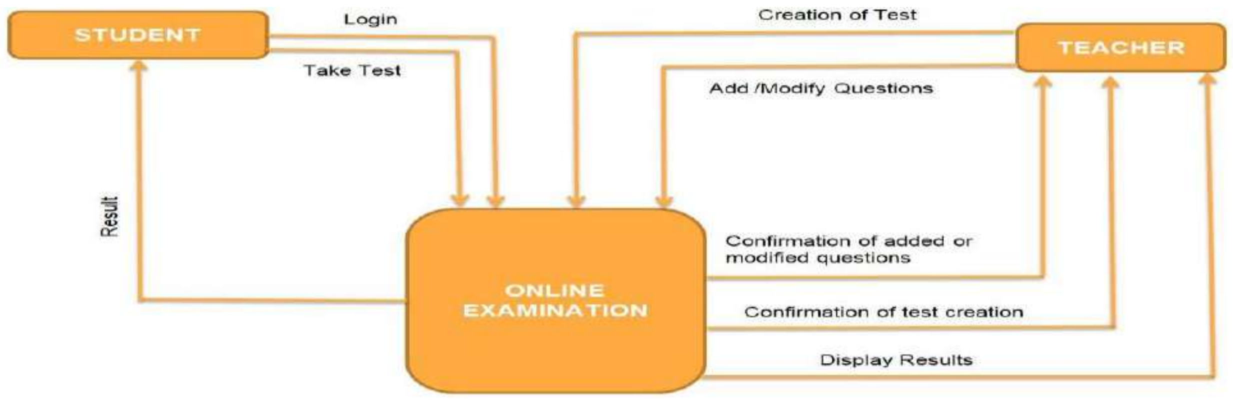


Fig 3.2

➢ Database (Backend)

Design Database design is concerned with how data is represented and stored within the system. The examination questions, answers and test details must be stored in a persistent way. Moreover, we need to keep information about the students. The system stores the above information in a MongoDB database server. Such database has been chosen since it makes it easy for developers to store structured or unstructured data. It uses a JSON-LIKE format to store documents.

➢ User Interface (Frontend)

Design Usability is the ease with which a user can learn to operate, prepare inputs for, and interpret outputs of system or component. This usability of a system is made less more or less stressful by the usability and complexity of the user interface. The user interface of a system is therefore the part of the system that the end user interacts with. User interface design is concerned with how users add information to the system and with how the system presents information back to them.

CHAPTER-4.Implementation

To develop this web application we have used Model-View Controller Architecture. It is a pattern that divides the code into three parts (Model, View, Controller) .This is done to distinguish internal representations of data from how data is presented to and accepted by users.

A) Models:

To develop this web application we have used MongoDB. We have Created two collections i.e devuser, devcreatetest inside exam directory and we have three schemas for these collections.

A Mongoose schema defines the structure of the document, default values, validators etc., whereas a Mongoose model provides an interface to the database for creating, querying, updating, deleting records.

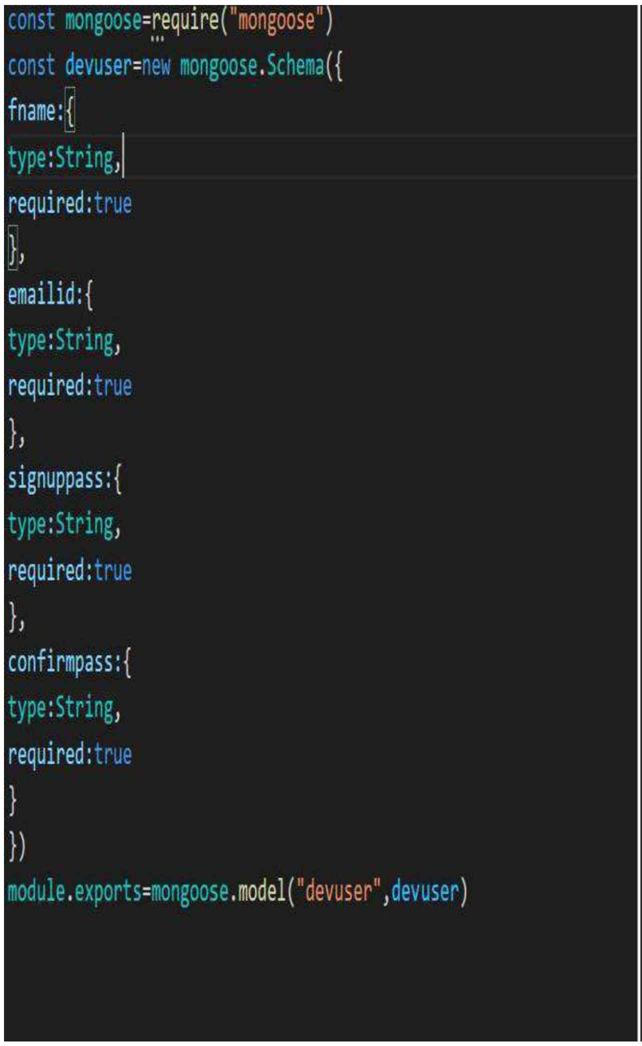
i) signupschema.js :

This mongoose schema consists of fname, emaild, signuppass, confirmpass. All the components are of String data type. This schema is used to structure and store the details of user, when the user signedin to the web application.

ii) createtestschema.js :

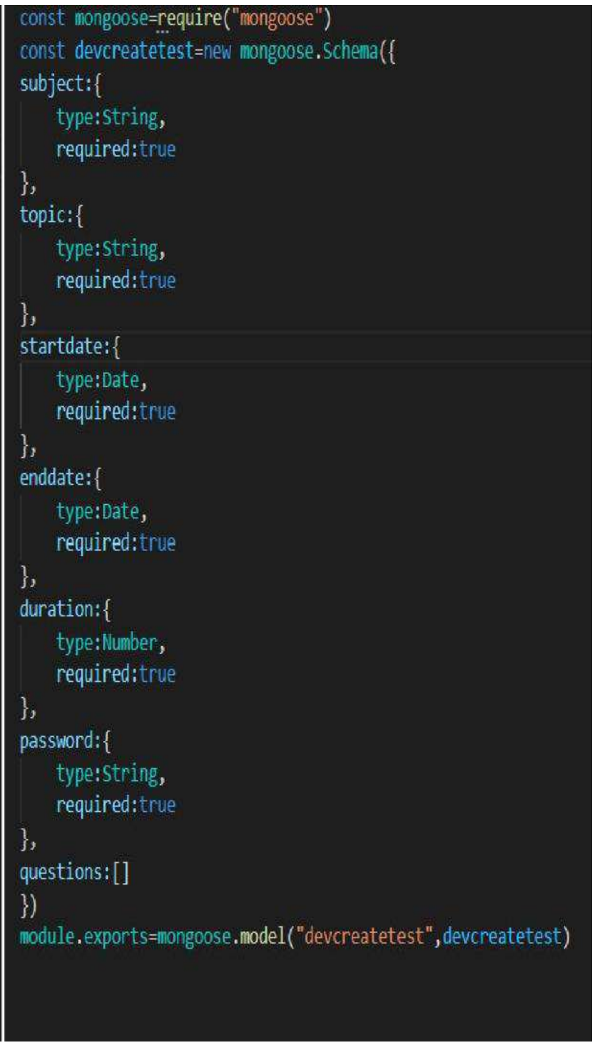
This mongoose schema is used to structure and store the test details when the teacher create test. This schema consists of subject, topic, startdate, enddate, duration, password, questions. The subject,topic are of String data type , startdate and enddate are of Date data type,duration is of Number data type, password is of String data type and and questionsare stored in array using questionschema.js

questionschema is created to structure the questions added by the teacher. This schema consisits of question, optionA, option,option, optionD, correctopt,marks. All the componenents are of String data types except marks which is of Number data type.



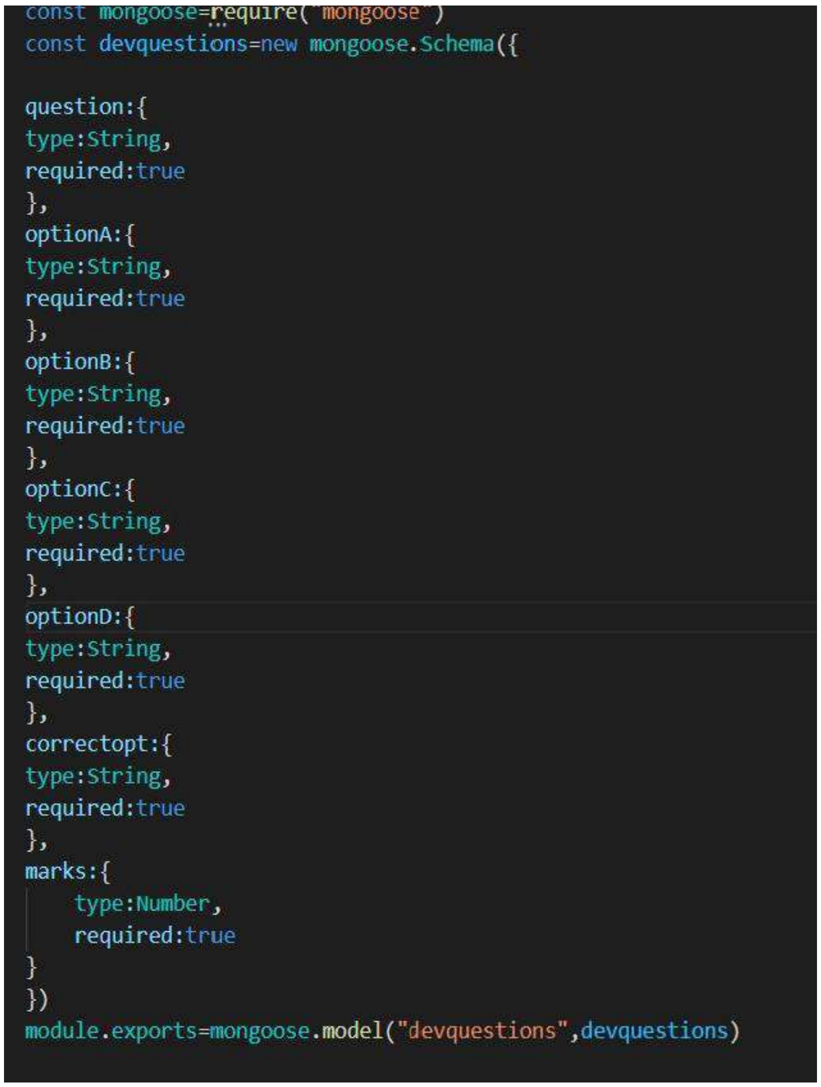
```
const mongoose=require("mongoose")
const devuser=new mongoose.Schema({
fname:{
type:String,
required:true
},
emailid:{
type:String,
required:true
},
signuppass:{
type:String,
required:true
},
confirmpass:{
type:String,
required:true
}
})
module.exports=mongoose.model("devuser",devuser)
```

Fig 4.1- Signup schema



```
const mongoose=require("mongoose")
const devcreatetest=new mongoose.Schema({
subject:{
    type:String,
    required:true
},
topic:{
    type:String,
    required:true
},
startdate:{
    type:Date,
    required:true
},
enddate:{
    type:Date,
    required:true
},
duration:{
    type:Number,
    required:true
},
password:{
    type:String,
    required:true
},
questions:[]
})
module.exports=mongoose.model("devcreatetest",devcreatetest)
```

Fig 4.2- create test schema



```
const mongoose=require("mongoose")
const devquestions=new mongoose.Schema({

question:{
type:String,
required:true
},
optionA:{
type:String,
required:true
},
optionB:{
type:String,
required:true
},
optionC:{
type:String,
required:true
},
optionD:{
type:String,
required:true
},
correctopt:{
type:String,
required:true
},
marks:{
    type:Number,
    required:true
}
})
module.exports=mongoose.model("devquestions",devquestions)
```

Fig 4.3- Questions schema

B) Views

Views are responsible for rendering the content in the web page by retrieving the data from database . For the front end, we used HTML,CSS and for the View engine, we used Express js. Viewengine is a programming language that integrates HTML elements, Server Controls, and a programming language in order to render views to the browser or the user within an application.

We have used following the following .ejs files to render the data from the database:

Student side:

i) **studentdash.ejs** - It was created to render the username of the student when the teacher login/signup to website.

ii) **instructions.ejs** - It was created to render the test details (subject,topic,start date ,enddate,duration) when the student entered the credentials to login to test.

iii) **mcq.ejs -**It was created to render the question details(question,options,marks) of particular test created by teacher using id of test.

Teacher side:

i) teacherdash.ejs: It was created to render the username of the Teacher when the teacher login/signup to website.

ii) teacherpreview.ejs: This file was created to render the questions details (questions,options,correct option,marks) so that teacher can preview the questions added and can modify and submit test.

C) Controller:

Controller is the main component in MVC architecture which communicates between model and views.

i) app.js- This file controls the model and views using get,post requests. All the get or post requests like userlogin,testlogin,createtest,etc are written here to retrieve the data from model and render it to view.

ii) face-api.min.js -It was created to load the face API model and proctor the examination while the student start the test. This controls the proctoring of examination and warns the student when ever the student try to manipulate.

CHAPTER-5.Results

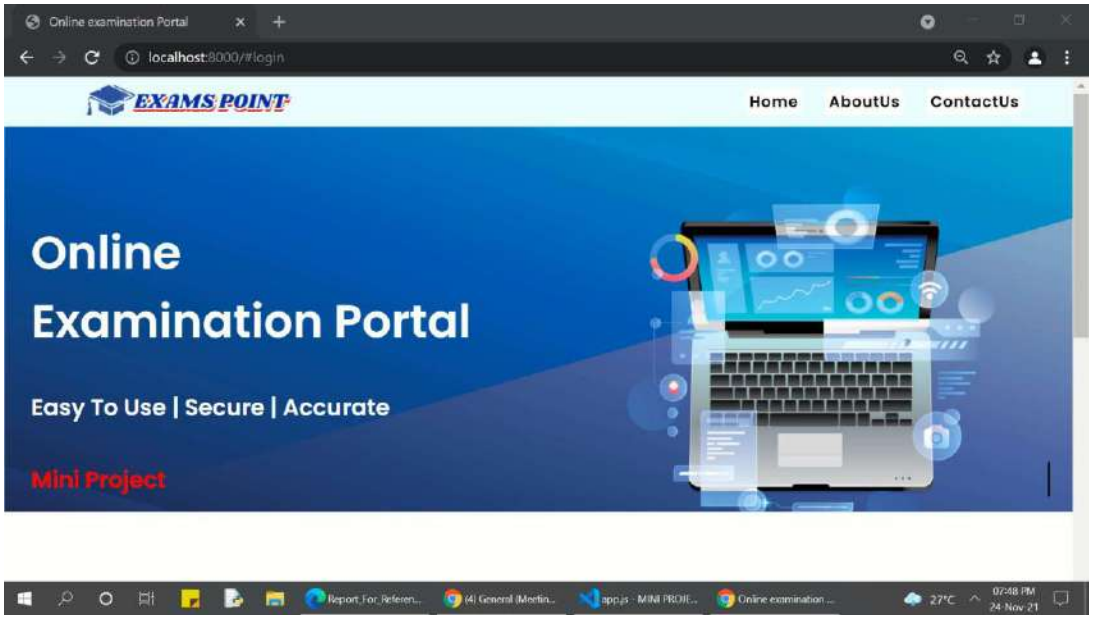


Fig 5.1 - Home page

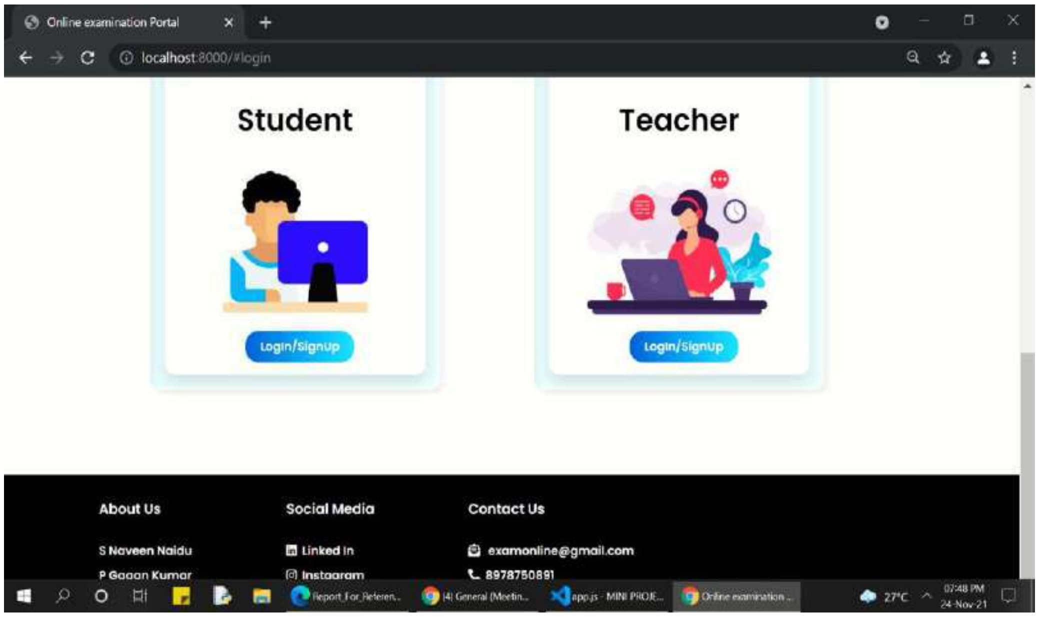
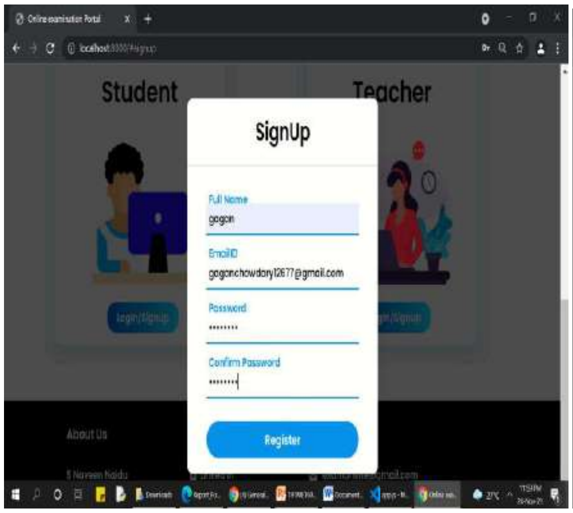


Fig 5.2 – Contact Page



Full Name: gagan

EmailID: gaganchowdary12677@gmail.com

Password: ••••••••

Confirm Password: ••••••••

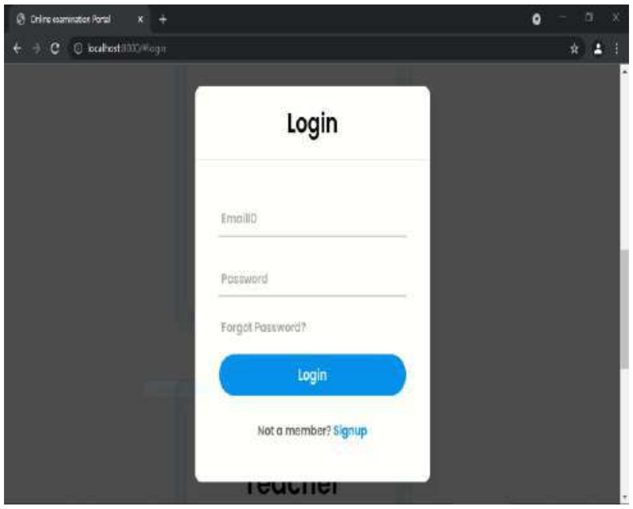


Fig 5.3 - Sign up **Fig 5.4 - Login**

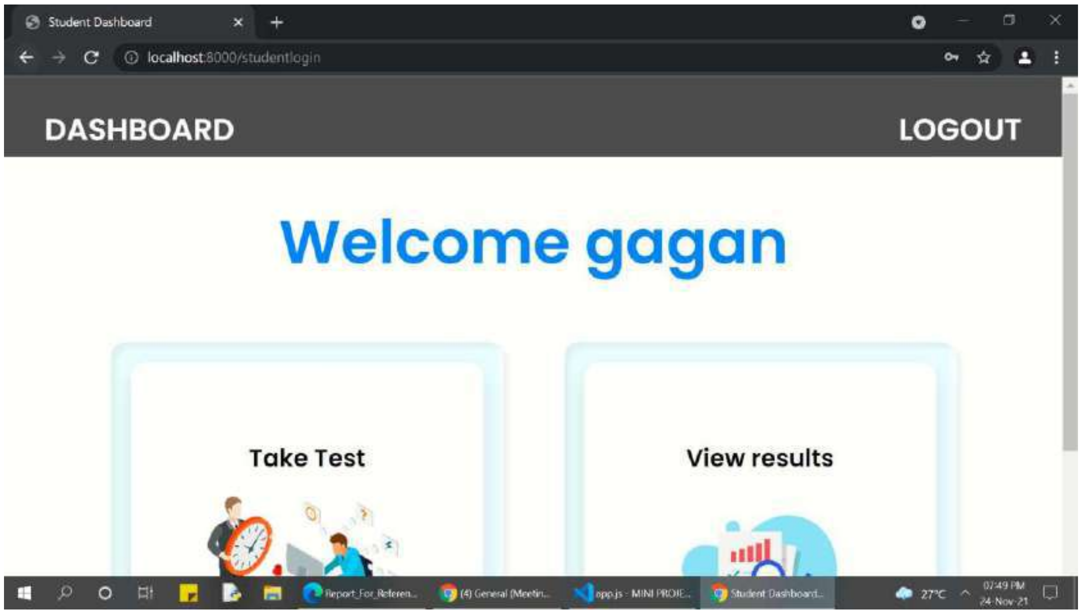


Fig 5.5 - Student Dashboard

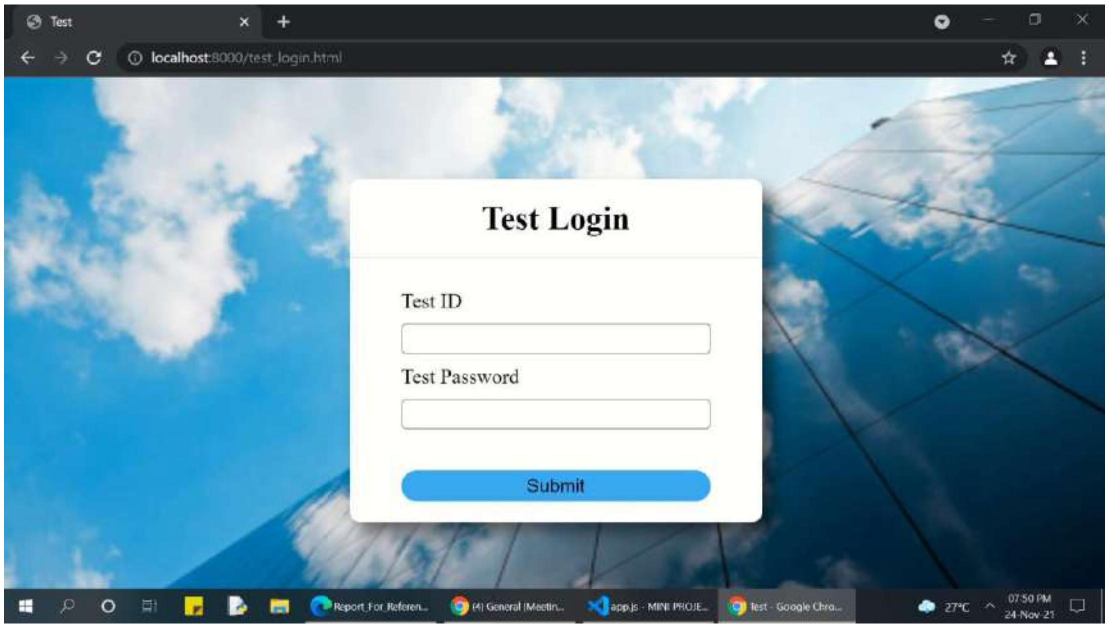
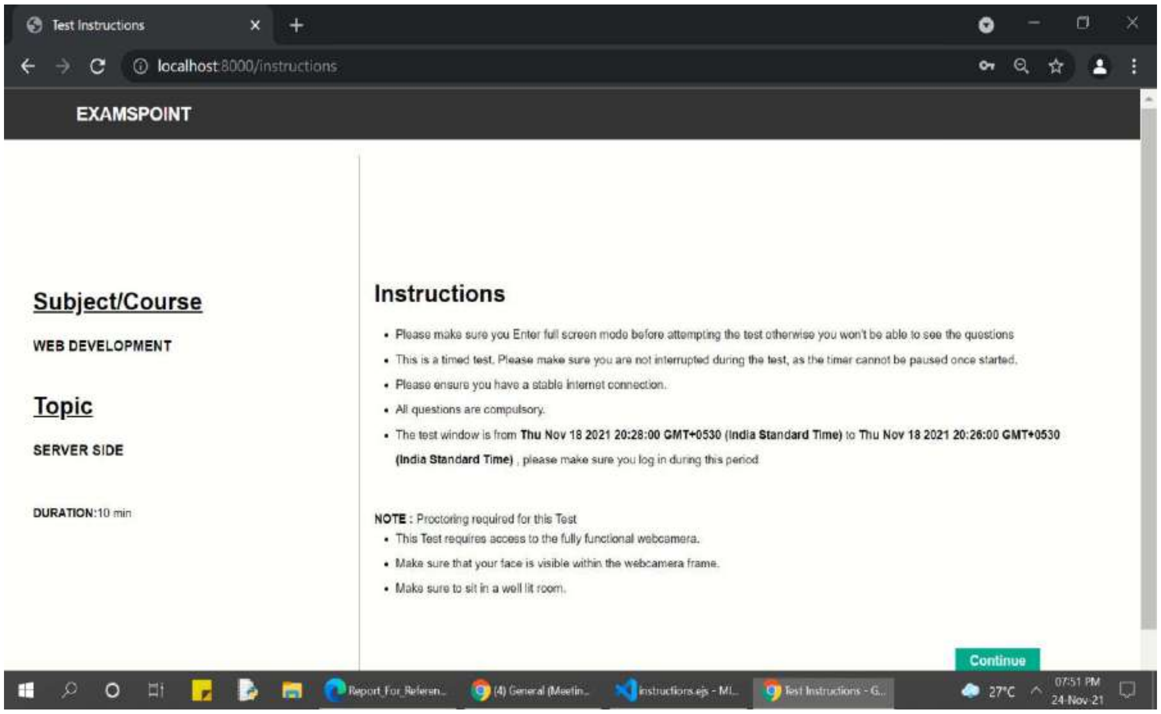


Fig 5.6-Test Login



Subject/Course

WEB DEVELOPMENT

Topic

SERVER SIDE

DURATION:10 min

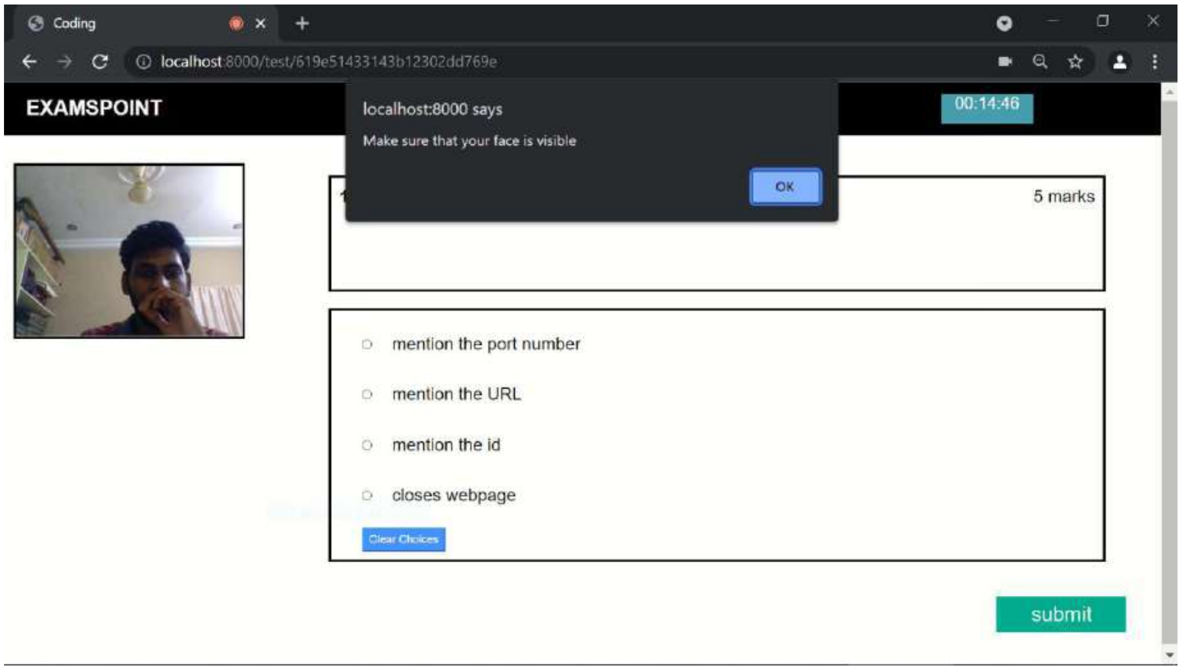
Instructions

- Please make sure you Enter full screen mode before attempting the test otherwise you won't be able to see the questions
- This is a timed test. Please make sure you are not interrupted during the test, as the timer cannot be paused once started.
- Please ensure you have a stable internet connection.
- All questions are compulsory.
- The test window is from **Thu Nov 18 2021 20:28:00 GMT+0530 (India Standard Time)** to **Thu Nov 18 2021 20:26:00 GMT+0530 (India Standard Time)** , please make sure you log in during this period

NOTE : Proctoring required for this Test

- This Test requires access to the fully functional webcamera.
- Make sure that your face is visible within the webcamera frame.
- Make sure to sit in a well lit room.

Fig 5.7- Instructions



localhost:8000 says

Make sure that your face is visible

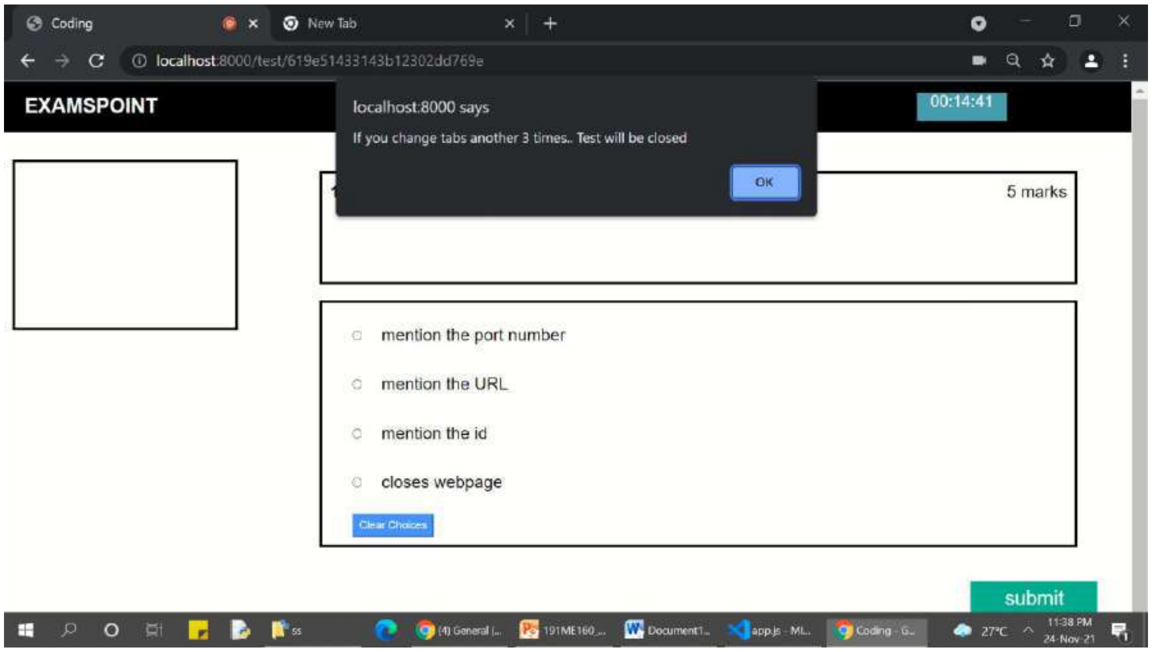
mention the port number

mention the URL

mention the id

closes webpage

Fig 5.8 - Proctoring – Face not visible



localhost:8000 says

If you change tabs another 3 times.. Test will be closed

mention the port number

mention the URL

mention the id

closes webpage

Fig 5.9 - Proctoring – when tab change

Teachers Side :

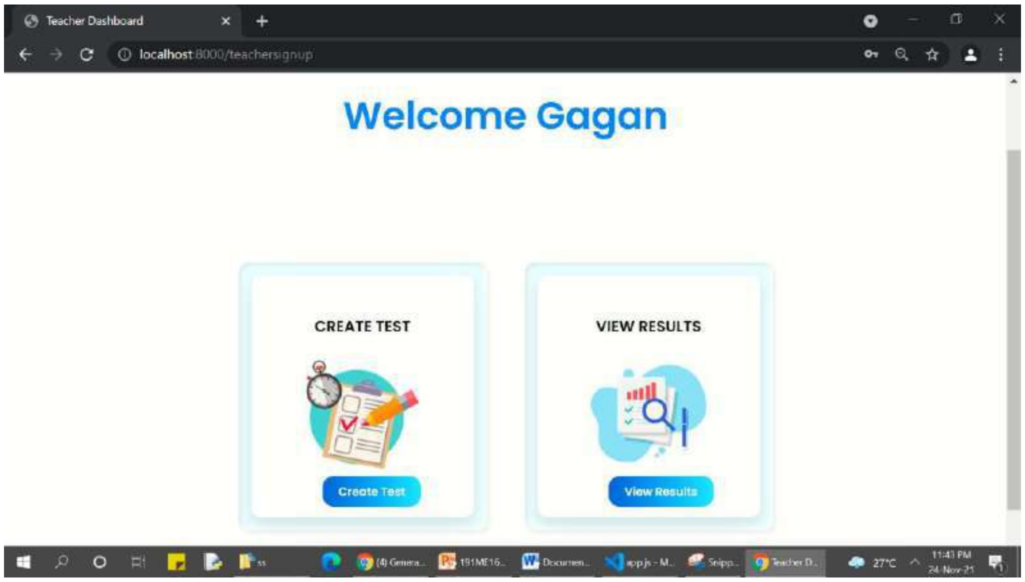
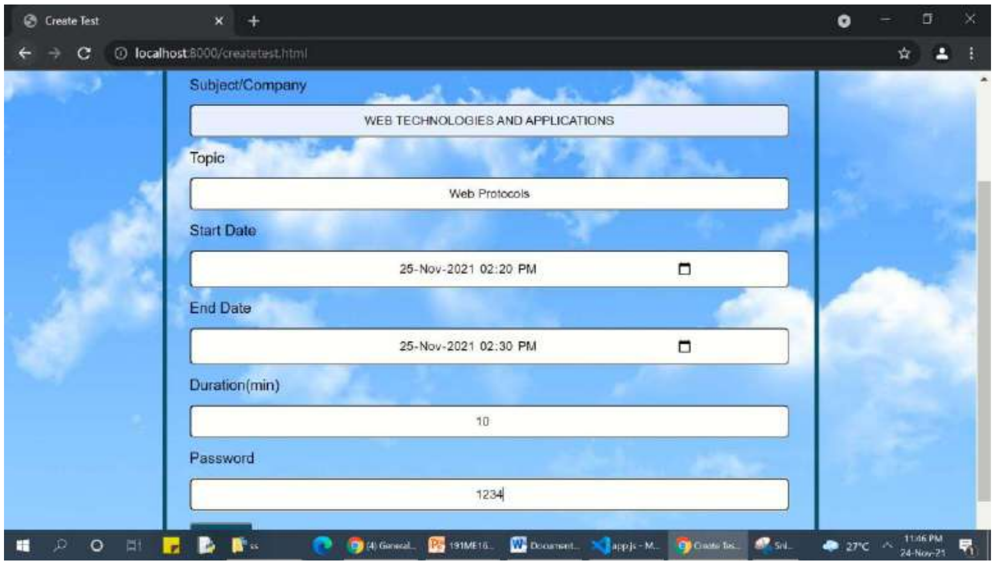


Fig 5.10- Teacher Dashboard



Subject/Company

WEB TECHNOLOGIES AND APPLICATIONS

Topic

Web Protocols

Start Date

25-Nov-2021 02:20 PM

End Date

25-Nov-2021 02:30 PM

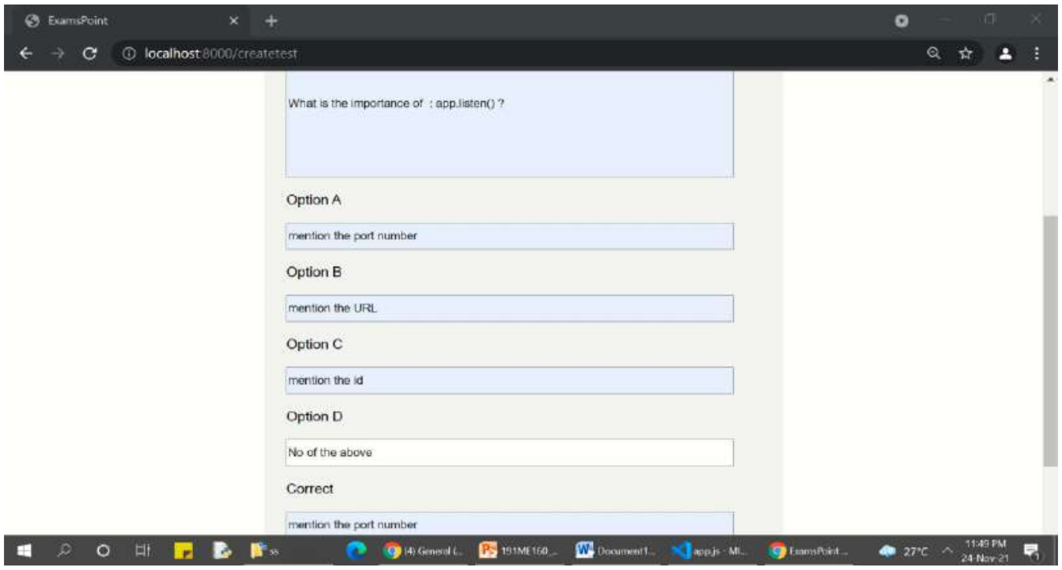
Duration(min)

10

Password

1234

Fig 5.11- Create Test



What is the importance of : app.listen() ?

Option A

mention the port number

Option B

mention the URL

Option C

mention the id

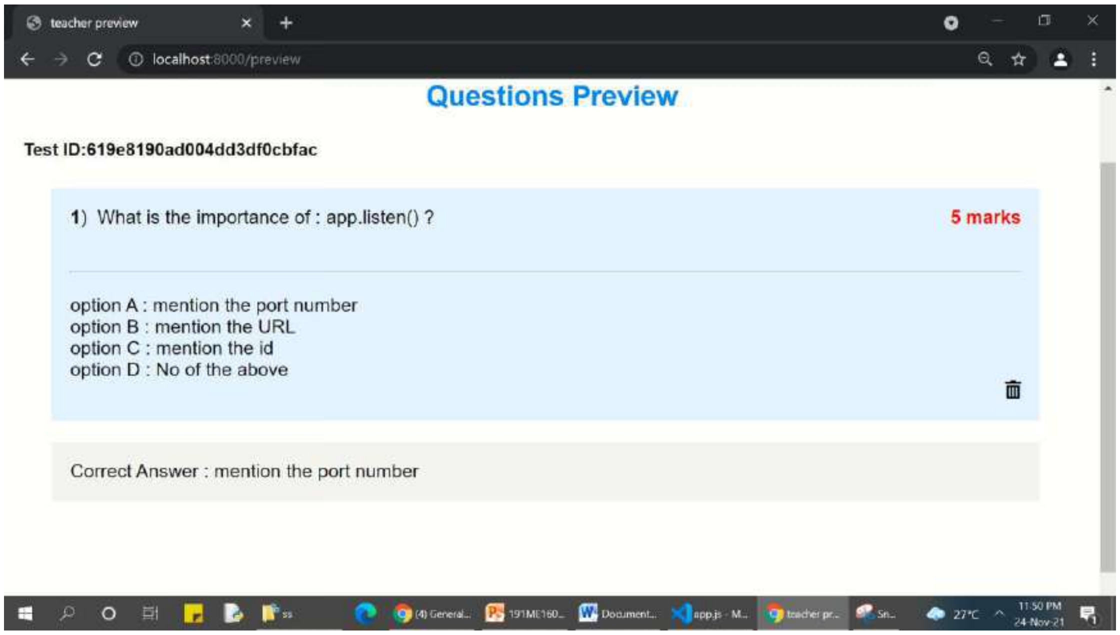
Option D

No of the above

Correct

mention the port number

Fig 5.12- Adding Questions



Questions Preview

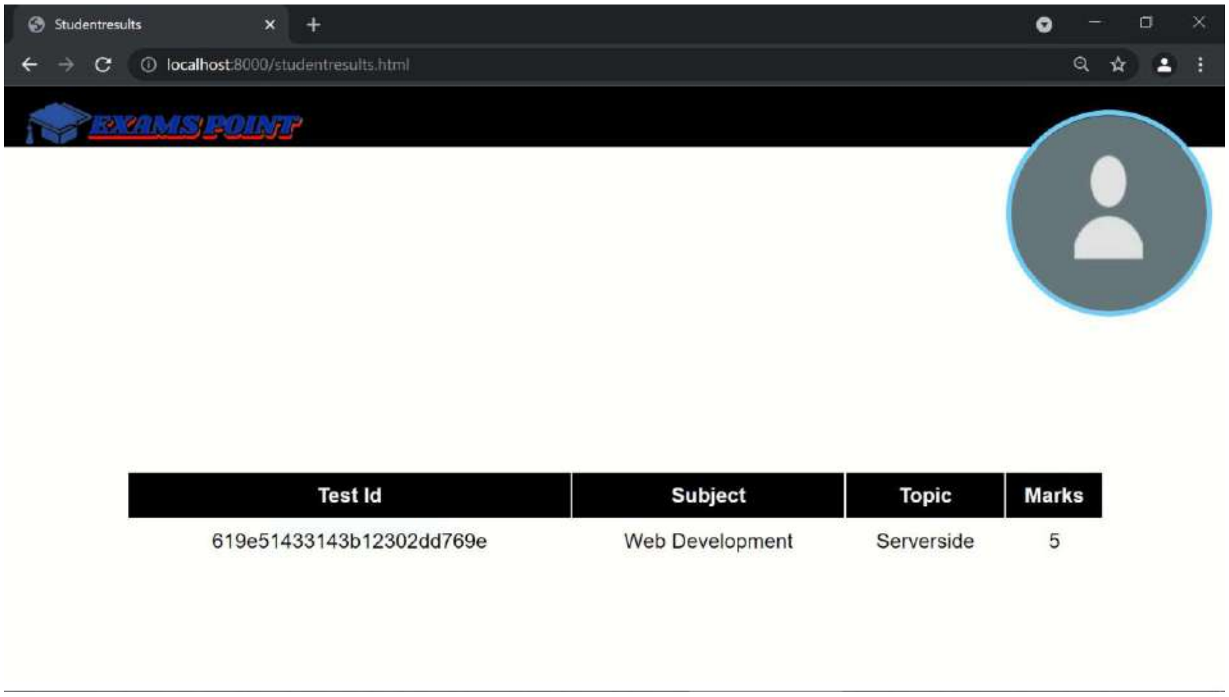
Test ID:619e8190ad004dd3df0cbfac

1) What is the importance of : app.listen() ? 5 marks

option A : mention the port number
option B : mention the URL
option C : mention the id
option D : No of the above

Correct Answer : mention the port number

Fig 5.13- Questions Preview


Studentresults

localhost:8000/studentresults.html

Test Id	Subject	Topic	Marks
619e51433143b12302dd769e	Web Development	Serverside	5

Fig 5.14- Student Results

CHAPTER-6.Work Plan & Future Plan

Work Plan:

- At first we designed dashboards for students, teachers
- Designed front end part of tests ,test logins, results page .
- Developed back end part and connected to database
- Tested the web application

Future Work :

- In future we are planning to include compiler for automata type questions and also files uploads for uploading the scanned pdfs .
- Improving proctoring features to minimize malpractices to the best.
- Increasing security .

CHAPTER-7.Conclusion

- This web application provides facilities to take online examination in a safe and secure mode from anywhere.
- This application will reduce malpractices to its best.
- Reduces time as it displays results immediately after the completion of the exam.
- In these pandemic times most of the organizations are willing to conduct their examinations in online mode. This web application might fulfill their requirements.

CHAPTER-8.References

- https://www.elearning-conf.org/wp-content/uploads/2020/07/03_202007L010_F095.pdf
- https://deliverypdf.ssrn.com/delivery.php?ID=574084113083007007097121094006021088050082052006043055030097112006006010106090103000029045032055014058032023029075091076003029033070056089028114127022011022081074030063012064091083091096072127002121021004071124118027117127019077019119064003028111102013&EXT=pdf&INDEX=TRUE