**COMPSOFT TECHNOLOGIES**



**AN INTERNSHIP REPORT**

**ON**

“FULL STACK WEB DEVELOPMENT”

##### **BACHELOR OF ENGINEERING**

**IN**

**COMPUTERSCIENCE & ENGINEERING**

***Submitted By:***

**PRABHAKAR BHIMANNA KATTIMANI(1SG19CS075)**

**SRISHTI SUMAN (1SG19CS107)**

**Department of ComputerScience and Engineering**

### SAPTHAGIRICOLLEGEOFENGINEERING

##### Affiliated to VTU, Belagavi, Approved by AICTE, NEW DELHI(ISO9001-2015 &ISO14001-2015 certified Institute)

***14/5,Chikkasandra,HesarghattaMain Road,Bengaluru–560057***

**2021-2022**

## ABOUT THE COMPANY

We are a digital service provider that aims to provide software, designing and marketing solutions to individuals and businesses, we believe that service and quality is the key to success

We provide all kinds of technological and designing solutions from Billing Software to Web Designs or any custom demand that you may have. Experience the service like none other!

Some of our services include:

Development - We develop responsive, functional and super-fast websites.

We keep User Experience in mind while creating websites. A website should load quickly and should be accessible even on a small view-port and slow internet connection.

Mobile Application - We offer a wide range of professional Android, iOS & Hybrid app development services for our global clients, from a startup to a large enterprise.

Design - We offer professional Graphic design, Brochure design & Logo design. We are experts in crafting visual content to convey the right message to the customers.

Consultancy - We are here to provide you with expert advice on your design and development requirement.

Videos - We create a polished professional video that impresses your audience.

**TABLE OF CONTENTS**

**Tableofcontents i**

[**Overview ofthe Project 1**](#_TOC_250006)

[**Implementation**](#_TOC_250003) **2**

[**Tools Used**](#_TOC_250002) **3**

**SourceCode 5**

[**Snapshots 17**](#_TOC_250001)

[**Bibliography 28**](#_TOC_250000)

## OVERVIEW OF THE PROJECT

## Project on FULL STACK WEB DEVELOPMENT

**Team Members:**1.**PRABHAKAR BHIMANNA KATTIMANI**

**2.SRISHTI SUMAN**

## This project is based on Web Development And its Applications. The main objective of this project is to learn the implementation of HTML, CSS and JavaScript. The basic webpage of this project is created using HTML and styling of the webpage is done using CSS.

**IMPLEMENTATION**

This chapter of the report describes the Functions, packages and modules used in the project:

**Libraries and Frameworks**

# PHP

# PHPisHypertext Pre-processor is a general-purpose programming language originally designed for web development.

# HTML

Hypertext Markup Language is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets and scripting languages such as JavaScript.

**CSS**

Cascading Style Sheets is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.FunctionalModules

**JavaScript**

# JavaScript, often abbreviated as JS, is a high-level, interpreted scripting language that conforms to the ECMAScript specification. JavaScript has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions.

**PHPMailer**

PHPMailer is a code library to send emails safely and easily via PHP code from a web server. Sending emails directly by PHP code requires a high-level familiarity to SMTP standard protocol and related issues and vulnerabilities about Email injection for spamming.

.

# TOOLS USED

### Software Requirements

* Visual Studio Code2019.
* Google Chrome or Microsoft Edge of latestversion.
* Front End: HTML,CSS, JS (optional)
* Linux 7.1 or Windows XP/7/8/10OS or Mac OS

### Hardware Requirements

* Pentium 200-MHz computer with a minimum of 64 MB of RAM (128 MB of RAM recommended).
* Monitor with a refresh rate of at least 40Hz for a smooth GUI experience(optional).

**SOURCE CODE:**

constdb = require("../models");

constjwt = require('jsonwebtoken');

constbcrypt = require("bcryptjs");

varvalidator = require("validator");

constpassport = require("passport");

constUtil = require("../util/Util");

constUser = {

  signIn:function (req, res,next ){

           returnpassport.authenticate("user\_local",{session:false},

            (err, user, info) => {

                if (err || !user) {

                    returnres.status(400).json({

                        status:"failed",

                        msg:info ? info.message :'Login failed',

                    });

                }

                req.login(user, {session:false}, (err) => {

                   if (err) {

                       res.status(500).json({status:"failed",msg:err});

                   }

                   //filtering user id and email for payload and setting exp time as 7 day

                   letpayload=JSON.stringify({"id":user.\_id,username:user.name,"email":user.email, exp:Math.floor(Date.now() / 1000) + (60 \* 60\*24\*7)});

                   // generate a signed json web token with the contents of user object and return it in the response

                   consttoken = jwt.sign(payload, process.env.JWT\_KEY);

                   res.json({status:"sucess",token});

                });

            })

      (req,res)

  },

  signUp:asyncfunction(req, res) {

    let {

        name,

        email,

        password,

    } = req.body;

    if (name&&email&&password ) {

        if (!validator.isEmail(email)) {

            res.json( {

                status:"failed",

                msg:"Invalid Email"

            });

            return

        }

        db.User.create({

            name,

            email,

            password

        })

        .then(asyncnew\_user=> {

            if (new\_user) {

              letlink = req.protocol + "://" + req.get("host") + "/user/verifiy/email/" + new\_user.\_id;

              letmsg =awaitUtil.verfiyMail(new\_user.email, new\_user.name, link);

              if (msg) {

                  res.json({"status":"sucess","msg":"Account created sucessfully"});

              } else {

                  //need to remove user from database  if mail not send sucessfully

                     db.User.deleteOne({

                          \_id:new\_user.\_id

                      })

                  res.json({status:"failed",

                          msg:"Sorry Something went wrong. Please try again"

                  });

              }

            }

            else{

                res.json({status:"failed",

                            msg:"Sorry Something went wrong. Please try again"

                    });

            }

        })

        .catch(err=>{

          letmsg=Util.dbErrorHandler(err)

          Util.logError(err.msg,err)

          res.json({status:"failed",msg:msg});

        })

    }

    else{

       res.json({status:"failed",msg:"Please enter all the detail's."});

    }

  },

  verifiyMyEmail:function(req,res) {

    db.User.findOneAndUpdate({\_id:req.params.userId},{isEmailVerified:true})

    .then((user)=>{

      if(user){

        res.json({status:"sucess",msg:"email verified sucessfully"})

      }else{

        res.json({status:"sucess",msg:"email verified failed"})

      }

    })

    .catch(err=>{

          Util.logError(err.msg,err)

          res.json({status:"failed",

                            msg:"Sorry Something went wrong. Please try again"

                    });

    })

  },

  getMyProfile:function (req,res){

      res.json({

        status:"sucess",

        name:req.user.name

    });

  },

  updateMyProfile:asyncfunction(req,res){

      if (req.body.name&&req.body.old\_password) {

            let {

                name,

                old\_password,

                new\_password

            } = req.body;

            letuser\_id = req.user.\_id;

            letuser = awaitdb.User.findOne({

                \_id:user\_id

            });

            if (user) {

                if (user.checkPassword(old\_password)) {

                    if (new\_password) {

                        user.name = name;

                        user.password = new\_password;

                    } else {

                        user.name = name;

                        user.password=old\_password;

                    }

                    user.save()

                    .then((user)=>{

                        if (user) {

                          res.json({

                              status:"sucess",

                              name:user.name,

                              msg:"sucessfully updated"

                          });

                        }

                        else {

                            res.json( {

                                status:"failed",

                                name:req.user.name,

                                msg:"Something went wrong"

                            });

                         }

                    })

                    .catch((err) => {

                        letmsg = Util.dbErrorHandler(err)

                        Util.logError(err.msg,err)

                        res.json({

                            status:"failed",

                            name:req.user.name,

                            msg:msg

                        });

                    });

                } else {

                    res.json( {

                        status:"failed",

                        name:req.user.name,

                        msg:"Password does not match"

                    });

                }

            }

        }

        else {

                    res.json( {

                        status:"failed",

                        name:req.user.name,

                        msg:"Password does not match"

                    });

        }

  },

  forgetMyPassword:asyncfunction (req, res) {

        if (req.body.email) {

            letemail = req.body.email;

          try{

                varuser = awaitdb.User.findOne({

                    email:email

                });

                if (user) {

                    lettoken = Util.generateToken();

                    letlink = req.protocol + "://" + req.get("host") + "/user/reset/password/" + token;

                    //we adding 20 mins to current date and converting in to mili sec

                    letpassword\_reset\_expires = Date.now() + 20 \* 60 \* 1000;

                    //updating the user token

                    letnew\_user = awaitdb.User.findOneAndUpdate({

                        \_id:user.\_id

                    }, {

                        passwordResetToken:token,

                        passwordResetExpires:password\_reset\_expires

                    });

                    //sending mail to user

                    letmsg = awaitUtil.sendPasswordReset(user.email, user.name, link);

                    //if msg send sucessfully

                    if (msg) {

                        res.json({

                            status:"sucess",

                            msg:"Check your mail to reset the password"

                        });

                    } else {

                        res.json({

                            status:"failed",

                            msg:"Sorry Something went wrong. Please try again"

                        });

                    }

                    return

                }

                res.json({

                    status:"failed",

                    msg:"No user exit with given gmail"

                })

          }

          catch(e){

              Util.logError(err.msg,err);

              res.json({status:"failed",msg:"Sorry Something went wrong. Please try again"});

          }

        }

    },

    resetMyPassword:asyncfunction (req, res) {

        letpassword\_reset\_token = req.body.passwordId;

        letnew\_password = req.body.password;

        if (password\_reset\_token&&new\_password) {

            //finding the user

            varuser = awaitdb.User.findOne({

                passwordResetToken:password\_reset\_token,

                passwordResetExpires: {

                    $gt:Date.now()

                }

            });

            if (user) {

                lethash = bcrypt.hashSync(new\_password, 10);

                letnew\_user = awaitdb.User.findOneAndUpdate({

                    \_id:user.\_id

                }, {

                    passwordResetToken:null,

                    password:hash

                });

                res.json({

                    status:"sucess",

                    msg:"Password Updated"

                });

            } else {

                res.json({

                    status:"failed",

                    msg:"Link Expires"

                });

            }

            return

        }

        res.status(400).json({

            status:"failed",

            msg:"Link not found"

        });

    },

    getMyQuestions:function(req,res){

        db.Question

        .find({userId:req.user.\_id})

        .then((questions\_obj)=>{

           questions\_obj.forEach(question=>{

             question.\_doc.user={name:req.user.name}

           })

           res.json({status:"sucess",questions:questions\_obj});

        })

        .catch((err)=>{

          Util.logError(err.msg,err);

          res.json({status:"failed",msg:"Something went wrong"});

        })

    },

    getMySortedQuestions:function(req,res){

        if(req.query.sortBy&&req.query.type){

          db.Question

            .find({userId:req.user.\_id})

            .sort({[req.query.sortBy]:parseInt(req.query.type)})

            .then((questions\_obj)=>{

                questions\_obj.forEach(question=>{

                 question.\_doc.user={name:req.user.name}

               })

               res.json({status:"sucess",questions:questions\_obj});

            })

            .catch((err)=>{

              Util.logError(err.msg,err);

              res.json({status:"failed",msg:"Something went wrong"});

            })

        }else{

              res.json({status:"failed",msg:"Query missing"});

        }

    },

    //single question for editing

    getMyQuestion:function(req,res){

        db.Question.findOne({userId:req.user.\_id,\_id:req.params.questionId})

        .then((question\_obj)=>{

              if(question\_obj){

                res.json({status:"sucess",question:question\_obj});

              }else{

               res.json({status:"failed",msg:"Something went wrong"});

              }

        })

        .catch((err)=>{

          Util.logError(err.msg,err);

          res.json({status:"failed",msg:"Something went wrong"});

        })

  },

  likeMyQuestion:function(req,res){

    if(req.params.questionId){

      db.Question

        .findOneAndUpdate({\_id:req.params.questionId,likes:{"$nin":[String(req.user.\_id)]}},

                        {"$push":{likes:[String(req.user.\_id)]}}

                      )

        .then((question\_obj)=>{

          if(!question\_obj){

            //if user already liked remove his id

             db.Question

               .findOneAndUpdate({\_id:req.params.questionId},

                        {"$pull":{

                                  likes:{ $in:[String(req.user.\_id)]}

                                 }

                        }

                      ) .then((question\_obj)=>{});

          }

          res.send()

        })

    }

  },

  addMyQuestion:function (req,res){

      if(req.body.topics&&req.body.question){

         db.Question

           .create({userId:req.user.\_id,topics:req.body.topics,desc:req.body.question})

           .then((question\_obj)=>{

                res.json({status:"sucess",msg:"sucessfully added your question"})

            })

           .catch(err=>{

              logError(err.msg,err)

              res.json({status:"failed",msg:"Sorry Something went wrong. Please try again"});

            })

      }else{

        res.json({status:"failed",msg:"Please fill all the data"});

      }

  },

  updateMyQuestion:function (req,res){

      if(req.body.question\_id&&req.body.topics&&req.body.question){

        db.Question

          .findOneAndUpdate({\_id:req.body.question\_id,userId:req.user.\_id,},{topics:req.body.topics,desc:req.body.question})

          .then((question\_obj)=>{

                res.json({status:"sucess",msg:"sucessfully updated your question"})

          })

          .catch((err)=>{

            res.json({status:"failed",msg:"Something went wrong"});

          })

      }else{

            res.json({status:"failed",msg:"Please filled all things"});

      }

  },

  deleteMyQuestion:function(req,res) {

      if(req.params.questionId){

        db.Question

        .findOneAndRemove({\_id:req.params.questionId})

        .then((question\_obj)=>{

              res.json({status:"sucess",msg:"sucessfully deleted your question"})

              //also remove all the answer

              db.Answer

                .deleteMany({questionId:req.params.questionId})

                .then((answer\_obj)=>{})

        })

        .catch((err)=>{

          Util.logError(err.msg,err);

          res.json({status:"failed",msg:"Something went wrong"});

        })

      }else{

            res.json({status:"failed",msg:"questionId missing"});

      }

  },

  getMyAnswers:function(req,res){

        db.Answer

        .find({userId:req.user.\_id})

        .then((answers\_obj)=>{

           answers\_obj.forEach(answer=>{

             answer.\_doc.user={name:req.user.name}

           })

           res.json({status:"sucess",answers:answers\_obj});

        })

        .catch((err)=>{

          Util.logError(err.msg,err);

          res.json({status:"failed",msg:"Something went wrong"});

        })

  },

  getMyAnswer:function(req,res){

    if(req.params.answerId){

        db.Answer.findOne({userId:req.user.\_id,\_id:req.params.answerId})

        .then((answer\_obj)=>{

              if(answer\_obj){

                res.json({status:"sucess",answer:answer\_obj});

              }else{

               res.json({status:"failed",msg:"Something went wrong"});

              }

        })

        .catch((err)=>{

          Util.logError(err.msg,err);

          res.json({status:"failed",msg:"Something went wrong"});

        })

    }else{

            res.json({status:"failed",msg:"answerId missing"});

    }

  },

  addMyAnswer:function (req,res){

      if(req.body.answer&&req.body.question\_id){

         db.Answer

           .create({userId:req.user.\_id,questionId:req.body.question\_id,answer:req.body.answer})

           .then((answer\_obj)=>{

              if(answer\_obj){

                res.json({status:"sucess",answer:{...answer\_obj.\_doc,user:{name:req.user.name}}})

              }else{

               res.json({status:"failed",msg:"Sorry Something went wrong. Please try again"});

              }

            })

           .catch(err=>{

              Util.logError(err.msg,err);

              res.json({status:"failed",msg:"Sorry Something went wrong. Please try again"});

            })

      }else{

        res.json({status:"failed",msg:"Please fill all the data"});

      }

  },

  likeMyAnswer:function(req,res){

    if(req.params.answerId){

      db.Answer

        .findOneAndUpdate({\_id:req.params.answerId,likes:{"$nin":[String(req.user.\_id)]}},

                        {"$push":{likes:[String(req.user.\_id)]}}

                      )

        .then((answer\_obj)=>{

          if(!answer\_obj){

            //if user already liked remove his id

             db.Answer

               .findOneAndUpdate({\_id:req.params.answerId},

                        {"$pull":{

                                  likes:{ $in:[String(req.user.\_id)]}

                                 }

                        }

                      ) .then((answer\_obj)=>{});

          }

          res.send()

        })

    }

  },

  updateMyAnswer:function (req,res){

      if(req.body.answer\_id  &&req.body.answer){

        db.Answer

          .findOneAndUpdate({\_id:req.body.answer\_id,userId:req.user.\_id,},{answer:req.body.answer})

          .then((answer\_obj)=>{

              if(answer\_obj){

                res.json({status:"sucess",msg:"sucessfully updated your answer"})

              }else{

                 res.json({status:"failed",msg:"Something went wrong"});

              }

          })

          .catch((err)=>{

            Util.logError(err.msg,err);

            res.json({status:"failed",msg:"Something went wrong"});

          })

      }else{

            res.json({status:"failed",msg:"Please filled all things"});

      }

  },

  deleteMyAnswer:function(req,res) {

      if(req.params.answerId){

        db.Answer

        .findOneAndRemove({\_id:req.params.answerId})

        .then((answer\_obj)=>{

              res.json({status:"sucess",msg:"sucessfully deleted your answer"})

        })

        .catch((err)=>{

          Util.logError(err.msg,err);

          res.json({status:"failed",msg:"Something went wrong"});

        })

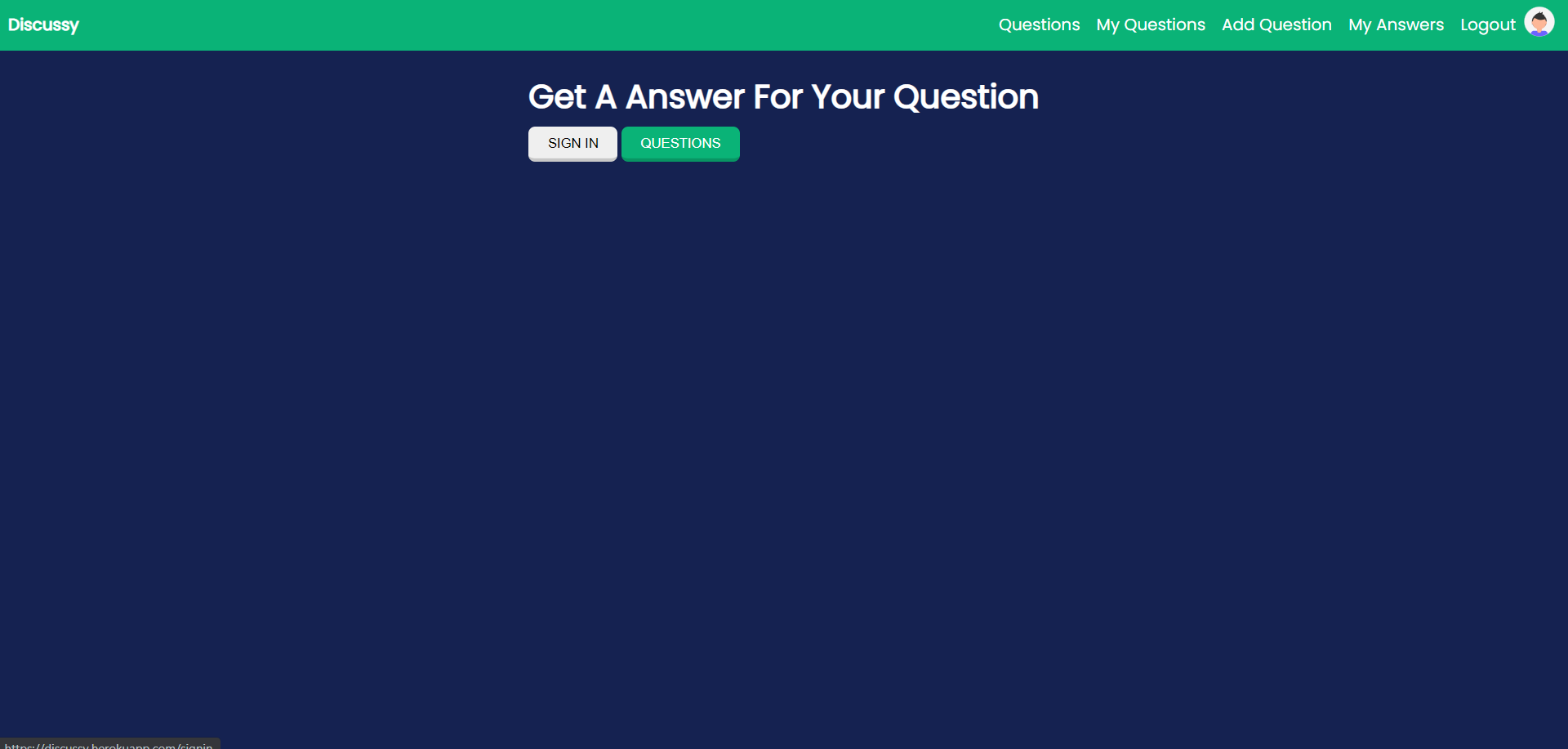
      }

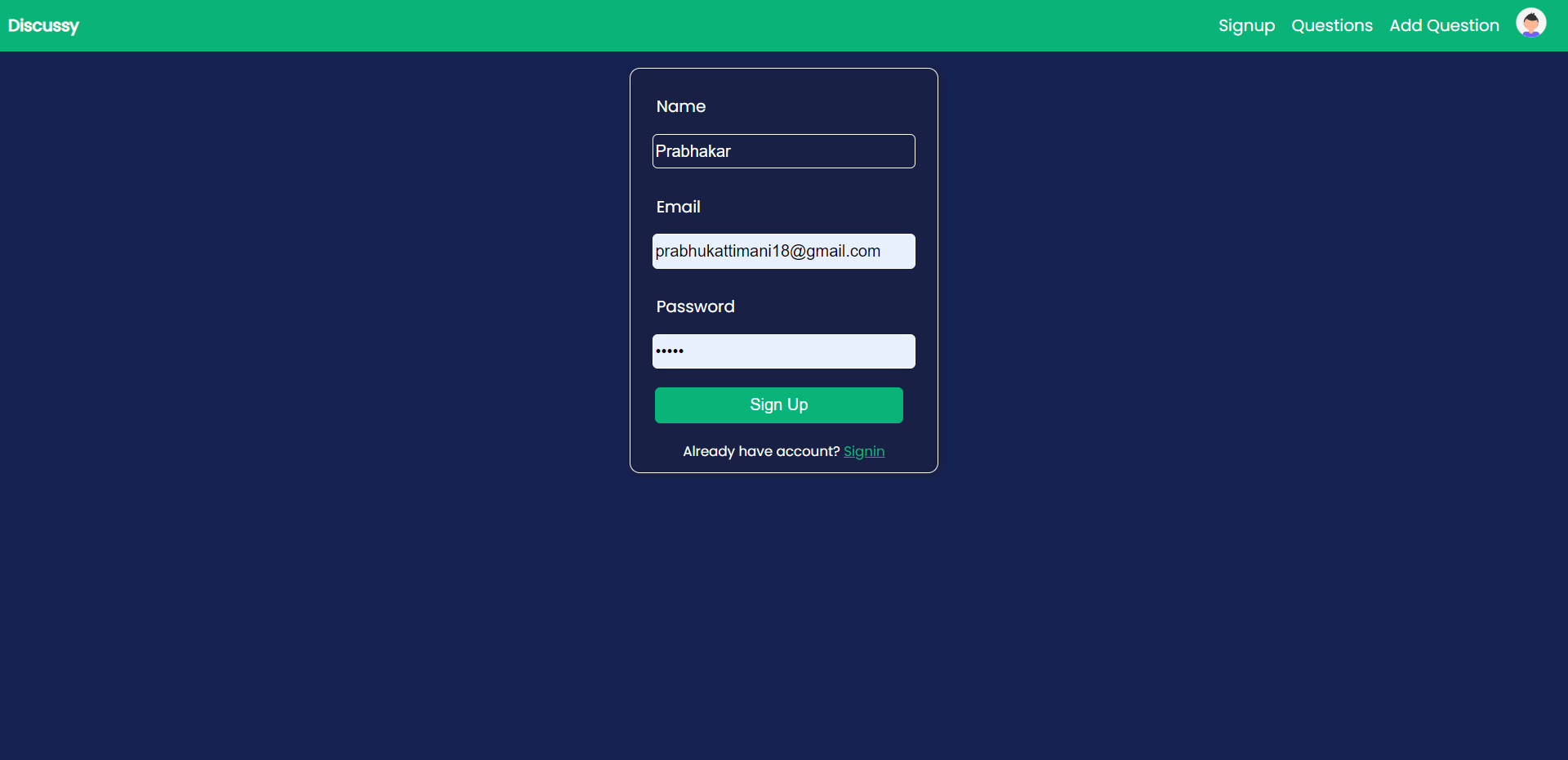
  },

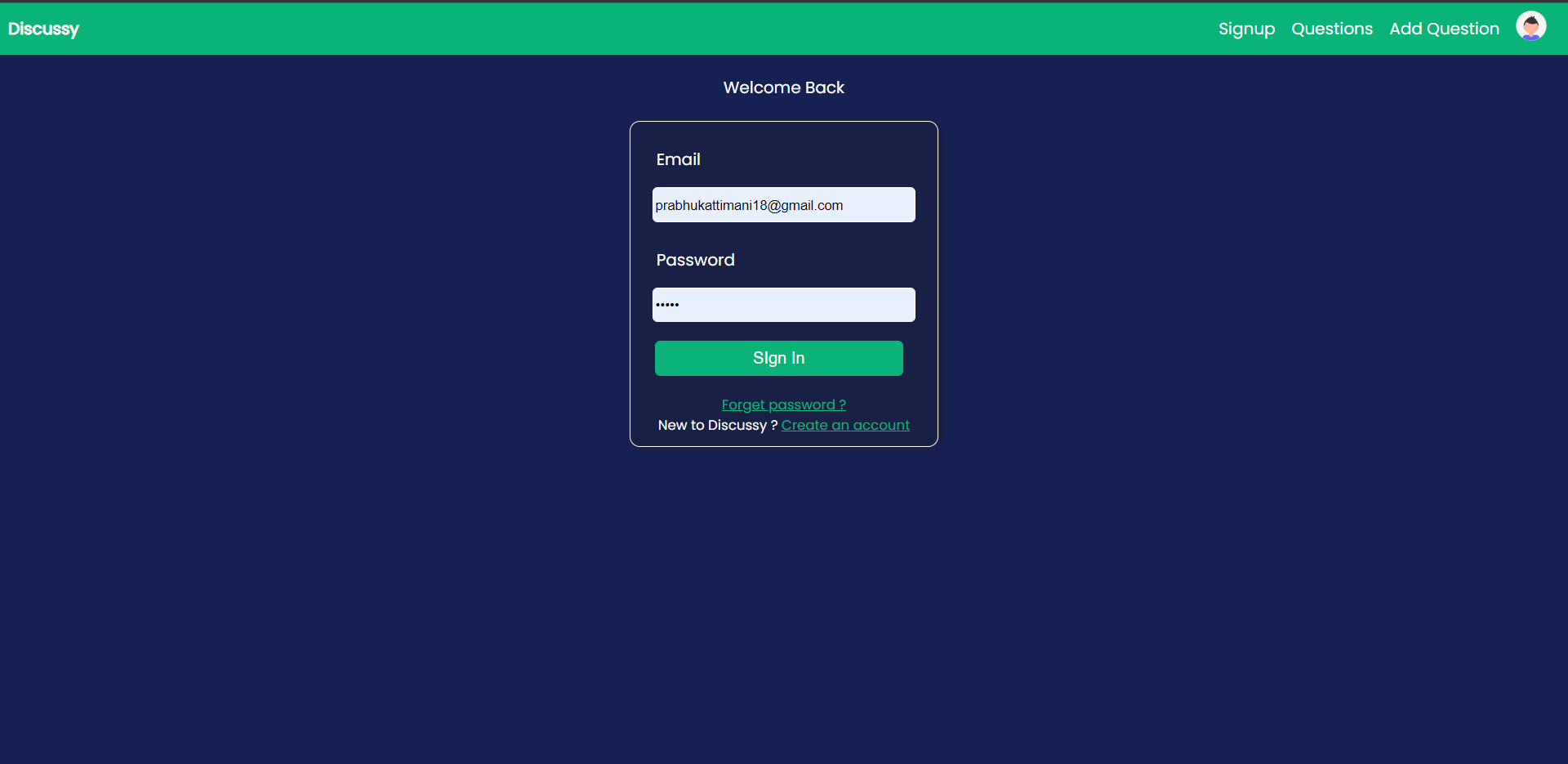
};

module.exports = User;

SNAPSHOTS :

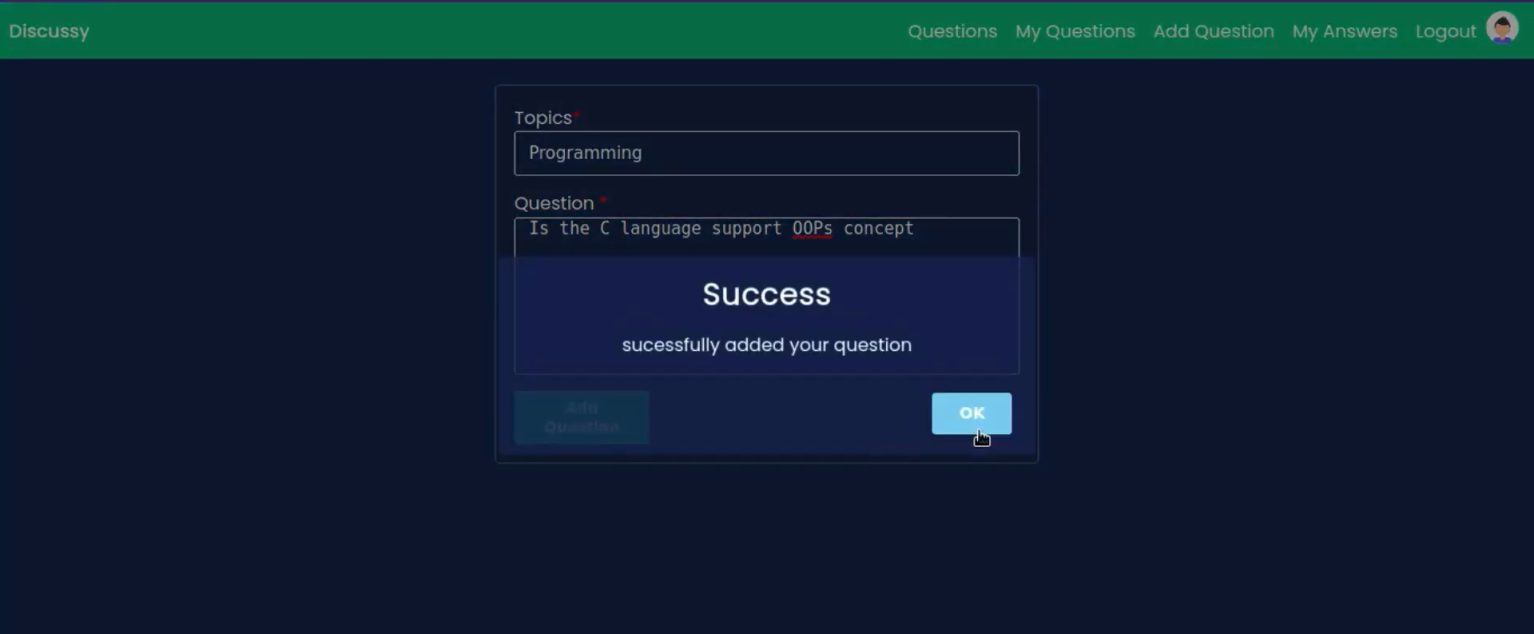


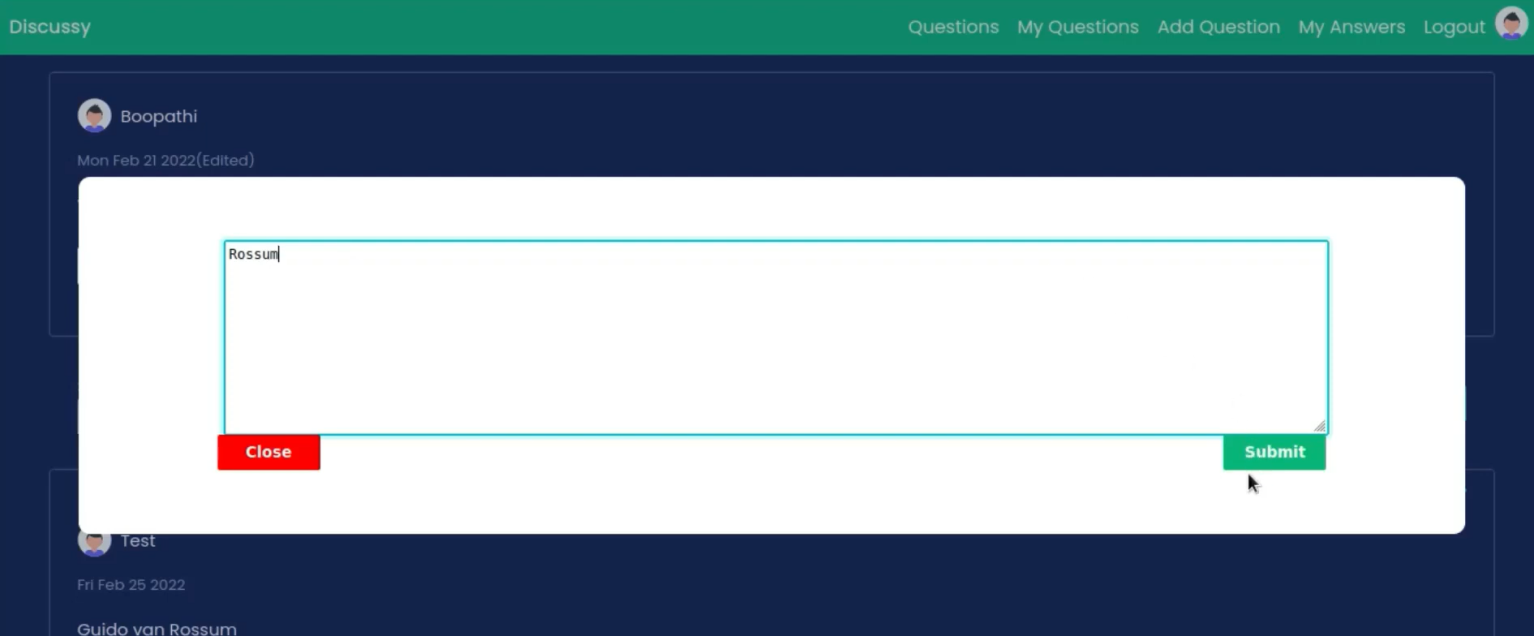


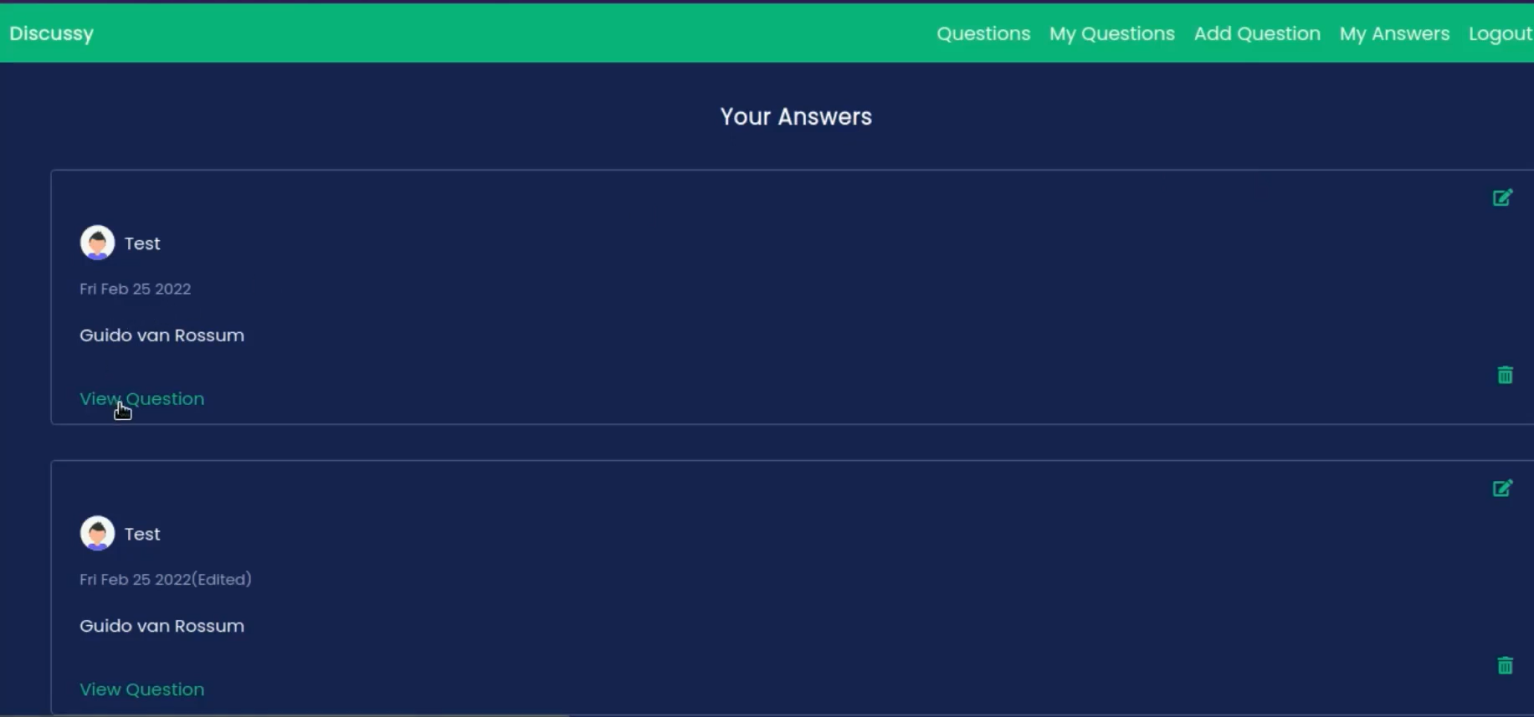


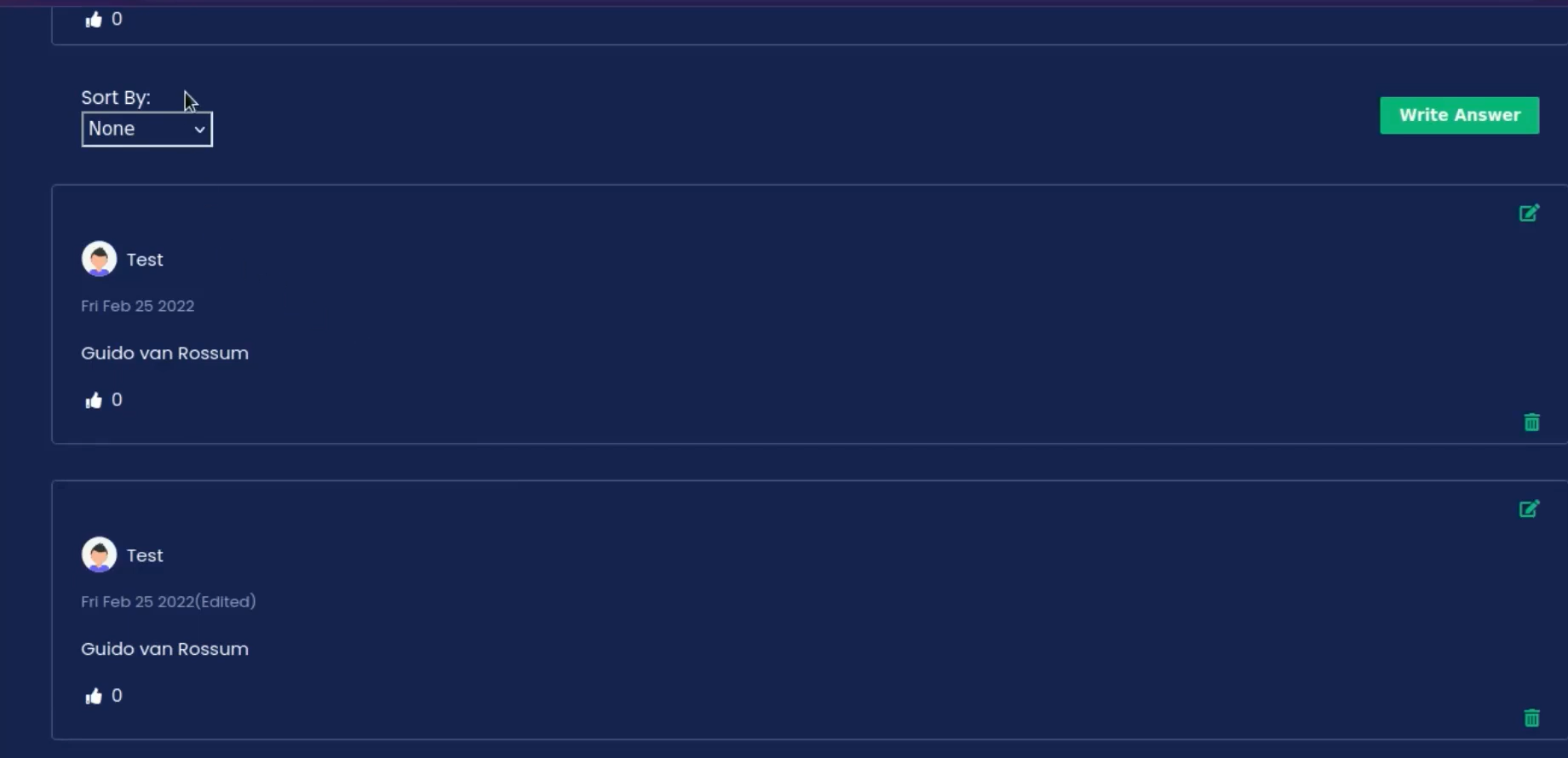
# C:\Users\srishti suman\OneDrive\Desktop\screanshots\page3.png

# C:\Users\srishti suman\OneDrive\Desktop\screanshots\page4.png



****







### BIBLIOGRAPHY

* + 1. Rakrishnan,&amp;Gehrke,J.
    2. Monson-Haefel,R.(2007).J2EEWebservices.Boston,Mass.:Addison-Wesley.Silberschatz,A.,Korth,H.F.,&amp;Sudarshan,S.(2011)
    3. Databasesystemsconcepts.EstadosUnidos:McGraw-HillCompanies,Inc.
    4. HannaP.(2002):JSP2.0TheCompleteReference,SecondEditionMcGrawHillEducation.

#### https://[www.w3schools.com/](http://www.w3schools.com/)php/default.asp

* + - * https://[www.php.net/](http://www.php.net/)

#### hellohttps://[www.apachefriends.org/download.html](http://www.apachefriends.org/download.html)