Dharmsinh Desai University, Nadiad

Faculty of Technology

Department of Computer Engineering

B. Tech. CE Semester – V

Subject: Advance Technology

Project title: <u>Online news portal,</u>

By:

1) **Charvit Ardeshna, roll no: CE009, Id: 19CEUOS130**

2) **Smit Bhoraniya, roll no: CE017, Id :19CEUOS057**

3) **Ishita Chauhan, roll no: CE021, Id: 19CEUOS140**

Guided by: Prashant M. Jadav

DHARMSINH DESAI UNIVERSITY
NADIAD-387001, GUJARAT

CERTIFICATE

This is to certify that the project entitled as "**<u>Online news portal</u>**" is a bonafide report of the work carried out by

1) **Charvit Ardeshna, roll no: CE009, Id: 19CEUOS130**

2) **Smit Bhoraniya, roll no: CE017, Id :19CEUOS057**

3) **Ishita Chauhan, roll no: CE021, Id: 19CEUOS140**

of Department of Computer Engineering, semester V, under the guidance and supervision of **Prof. Prashant M. Jadav** for the subject System Design Practice during the academic year 2019-2020.

Project Guide   Head of the Department

Assistance Professor   Dr. C.K Bhensdadia
Prof. Prashant M. Jadav   Prof. & Head,
Department of Computer   Department of Computer
Engineering,   Engineering,
Faculty of Technology,   Faculty of Technology,
Dharmsinh Desai University,   Dharmsinh Desai University,
Nadiad   Nadiad
Date: 24/10/2021   Date: 24/10/2021

Contents:

Abstract

Today the world totally relays upon the electronic media to its every day adventure. People have no time to be updated through newspaper or watching or listening the news on television or radios. People today need to be updated on daily basis in this competitive world. Most of the people get the information about the world around through the internet which is fast, accessible, and reliable. The WWW is huge, widely distributed, global information service centre for Information services: news, advertisements, consumer information, financial management, education, government, e-commerce etc, hyper-link information, access and usage information. "CSI NEWS Portal" is a service introduced to meet the above requirement and to make the people updated about the news, views, reviews, breaking news and latest headlines in different fields also the new inventions around the world.

Introduction

CSI NEWS Portal in which there are mainly two users Admin and Readers. This System is for people who wants to be updated on daily basis. Users needs to login to the system where they will find headlines, breaking news and news of category of their choice. Users can save the news and also, they can comment on news and view other users' comments. Users can modify their profile.

Admin can login to the system. Admin can Add news, Update News and Delete news.

Technologies/tools used :

- Platform used: Visual Studio Code
  Technology: MERN STACK
  Technology: WEB (application)
- Platform used: Mongodb Atlas (Online Database)

Software Requirement Specifications:

CSI News Portal

Modules:

- Admin module
- User module

User of the Online Railway System are:

- Admin:

  Admin of the system has power to maintain and update system.

  The admin can update database of the system, verify, update, delete news.

- Readers:

  Readers can register themselves, view, search, save news.

  Comment on news.

Details:

1.Admin module:

1.1 Verify admin:

- Description:

  Admin needs to login to manage website.

- Input:

  Information of admin.

- Output:

  Successfully logged in.

1.2 Add news:

- Description:

  Admin can add news to website by adding title, image, description and category of news.

- Input:

  Details of news.

- Output:

  Added news.

1.3 Modify news:

- Description:

  Admin can modify news to website by updating title, image, description and category of news.

- Input:

  Updated details of news.

- Output:

  Updated news.

1.3 Delete news:

- Description:

  Admin can Delete news to website.

- Input:

  Select news.

2. User module

2.1   Add user:

- Description:

  Users will have to enter their details username, email-id, password.

- Input:

  Information of users.

- Output:

  Successfully registered.

2.2 Modify user:

- Description:

  Users can modify their details like email-id, details.

- Input:

  Information user wants to modify.

- Output:

  Successfully modified.

2.3 Search news:

- Description:

  Users can search news by headlines or category.

- Input:

  Category, headline.

- Output:

  Available news.

2.4 Save news:

- Description:

  Users can save news of their choice.

- Input:

Select news.

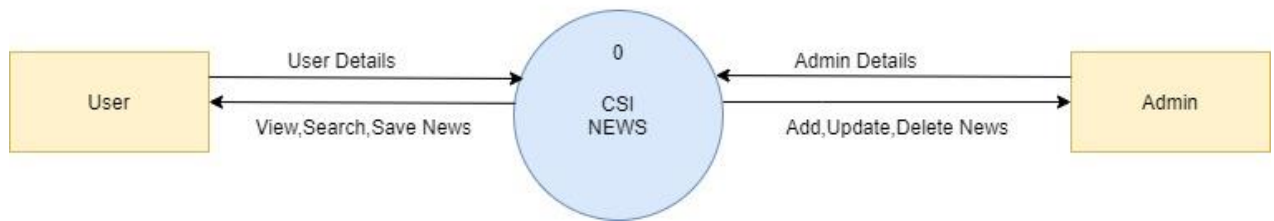- Output:

News will be saved to their profile.

2.5 Comments on news:

- Description:

Users can view and add comments to news

- Input:

Comment description.

- Output:

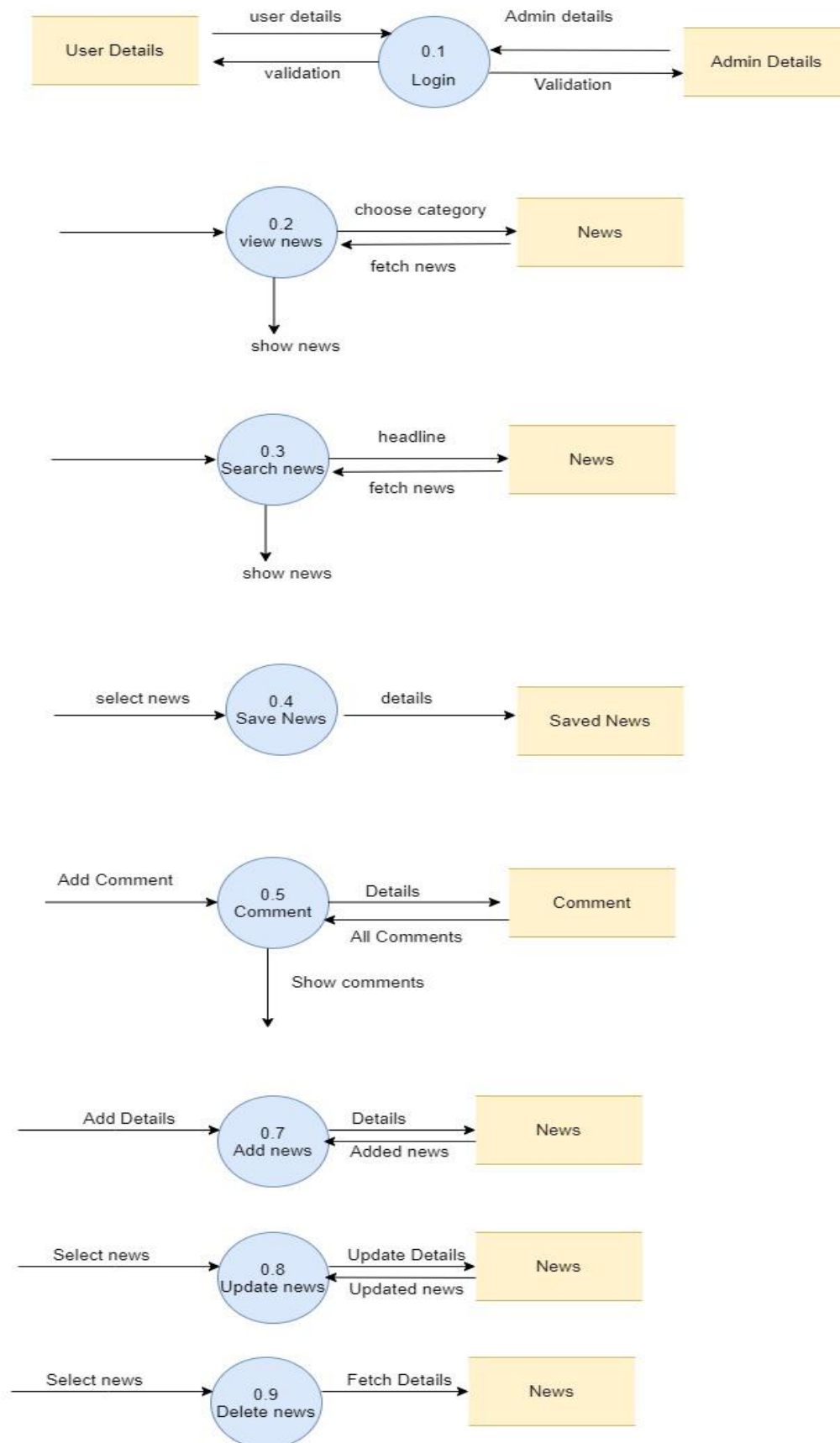Comments of all users on the same news.

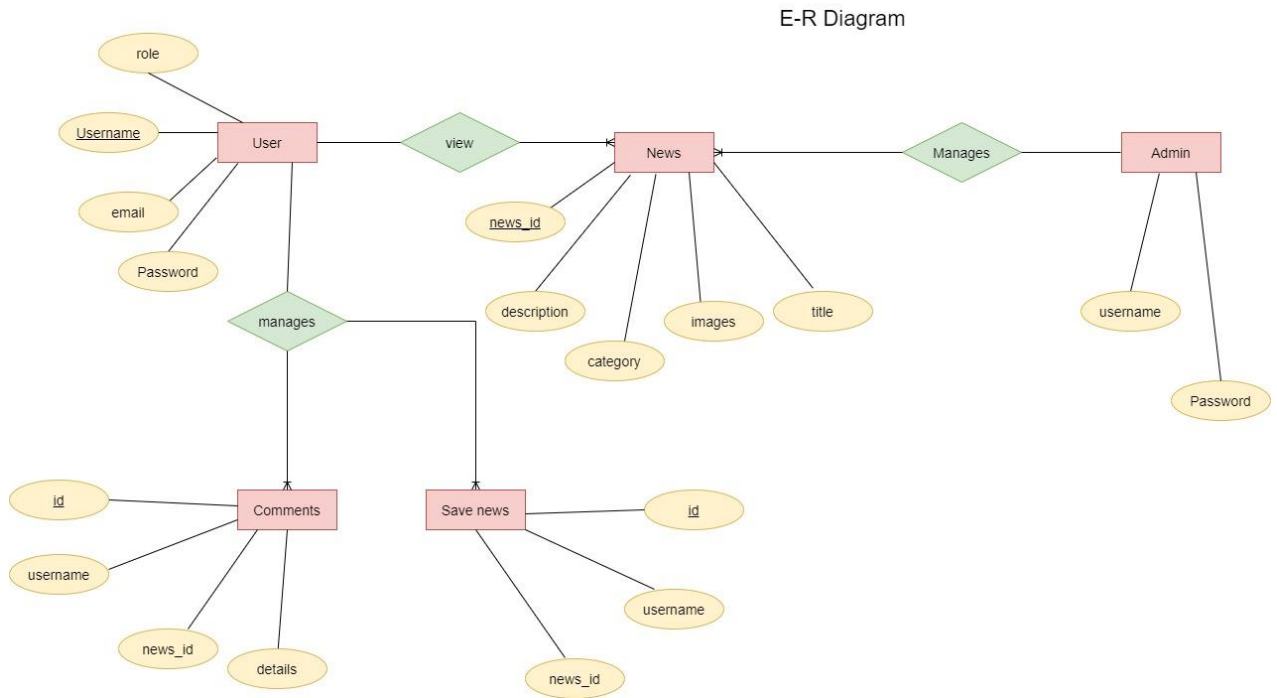# <u>Design:</u>

- Data flow diagram

CONTEXT DIAGRAM

User		CSI NEWS		Admin

User Details →

← View,Search,Save News

0
CSI
NEWS

← Admin Details

Add,Update,Delete News →

level-1

User Details →(user details)→ 0.1 Login ←(Admin details)← Admin Details

User Details ←(validation)← 0.1 Login →(Validation)→ Admin Details

→ 0.2 view news →(choose category)→ News

0.2 view news ←(fetch news)← News

0.2 view news ↓ show news

→ 0.3 Search news →(headline)→ News

0.3 Search news ←(fetch news)← News

0.3 Search news ↓ show news

(select news)→ 0.4 Save News →(details)→ Saved News

(Add Comment)→ 0.5 Comment →(Details)→ Comment

0.5 Comment ←(All Comments)← Comment

0.5 Comment ↓ Show comments

(Add Details)→ 0.7 Add news →(Details)→ News

0.7 Add news ←(Added news)← News

(Select news)→ 0.8 Update news →(Update Details)→ News

0.8 Update news ←(Updated news)← News

(Select news)→ 0.9 Delete news →(Fetch Details)→ News

10

- ER diagram

E-R Diagram

- ## Data Dictionary

  In our project we have used the Mongo dB database so we cannot create a data dictionary as well as entity relationship diagram but we are providing here some pictures of our database. We have used the mongo dB atlas. It doesn't have a table, row &document & field as shown in below pictures.

  UserSchema:

```javascript
const userSchema = new mongoose.Schema({
    username: {
        type: String,
        required: true,
        trim: true,
        maxlength: 30,
        unique: true
    },
    email: {
        type: String,
        required: true,
        trim: true,
        unique: true,
    },
    password: {
        type: String,
        required: true,
    },
    avatar: {
        type: String,
        default: 'https://i.pinimg.com/474x/65/25/a0/6525a08f1df98a2e3a545fe2ace4be47.jpg',
    },
    role: {
        type: String,
        default: 'user',
    },
    website: {
        type: String,
        default: '',
    },
    saved: [
        {
            type: mongoose.Types.ObjectId,
```

```
            ref: 'user'
        }
    ]
}, {
    timestamps: true
})

module.exports = mongoose.model('user', userSchema);
```

News Schema:

```
const newsSchema = new mongoose.Schema({
    title: String,
    images: {
        type: Array,
        required: true
    },
    content: String,
    category: String,
    likes: [{type: mongoose.Types.ObjectId, ref: 'user'}],
    comments: [{type: mongoose.Types.ObjectId, ref: 'comment'}],
    user: {type: mongoose.Types.ObjectId, ref: 'user'},
},{
    timestamps: true
});
```

Comment Schema:

```
const commentSchema = new mongoose.Schema({
    content: {
        type: String,
        required: true
    },
    user: {type: mongoose.Types.ObjectId, ref: 'user'},
    postId: mongoose.Types.ObjectId,
    postUserId: mongoose.Types.ObjectId
},{
    timestamps: true
});
```

Implementation Detail:

1. Modules:

 In the following section a brief description of each module is given. Related screenshots are attached in separate sections.

User-module:

  In User module user can view news of their choice. Users can search news by category and headline. Users can save news. Users can post comments on news and also they can view comments of other users.

Admin-module:

Admin modules appear in the admin side. Admin have all access to add news with their associated category. And can perform crud operations on news.

Authentication-module:

  Admin & User must have to login to the system in order to use any of the functionality. Detailed description can be seen in the screenshot section.

2. Major Functions prototypes

- **Login function:**

Admin and user need to provide their emailid and password. If it matches with the database then they can login.

```
login: async (req, res) => {
    try {
        const { email, password } = req.body;

        const user = await Users.findOne({ email });

        if (!user)
            return res.status(500).json({ msg: "This email does not exist." });

        const isMatch = await bcrypt.compare(password, user.password);

        if (!isMatch)
            return res.status(500).json({ msg: "Password is Wrong." });

        const access_token = createAccessToken({ id: user._id });
        const refresh_token = createRefreshToken({ id: user._id });

        res.cookie('refreshtoken', refresh_token, {
            httpOnly: true,
            path: '/api/refresh_token',
            maxAge: 30 * 7 * 24 * 60 * 60 * 1000 // 30days
        });

        res.json({
            msg: 'Login Successfully :)',
            access_token,
            user: {
                ...user._doc,
                password: ''
            }
        });
    } catch (err) {
        return res.status(500).json({msg: err.message});
    }
},
logout: async (req, res) => {
    try {
        res.clearCookie('refreshtoken', { path: '/api/refresh_token' })
        return res.json({ msg: "Logout Successfully!" })
    } catch (err) {
        return res.status(500).json({msg: err.message});
```

```
        }
    },
    generateAccessToken: async (req, res) => {
        try {
            const rf_token = req.cookies.refreshtoken;

            if (!rf_token)
                return res.status(400).json({ msg: "Please Login!!" });

            jwt.verify(rf_token, process.env.REFRESH_TOKEN_SECRET, async (err, result) => {
                if (err)
                    return res.status(400).json({ msg: "Please login!!" });

                const user = await Users.findById(result.id).select("-password");

                if (!user)
                    return res.status(400).json({ msg: "User is not exist :(" });

                const access_token = createAccessToken({ id: result.id });

                res.json({
                    access_token,
                    user
                });
            })
        } catch (err) {
            return res.status(500).json({msg: err.message});
        }
    }
}
};
```

- **Register function:**

    If user is new to the system he needs to fill the registration form and details of the form will be stored to the database.

```
register: async (req, res) => {
    try {
        const { username, email, password } = req.body;
        let newUserName = username.toLowerCase().replace(/ /g, '');

        if(!username || !email || !password)
```

```javascript
        return res.status(400).json({ msg: "Please fill all fields!!" });

    const user_name = await Users.findOne({ username: newUserName });
    if (user_name)
        return res.status(400).json({ msg: "This Username is already in Use." });

    const user_email = await Users.findOne({ email });
    if (user_email)
        return res.status(400).json({ msg: "This Email is already in Use." });

    if(!validateEmail(email))
        return res.status(400).json({ msg: "Invalid emails!!" });

    if (password.length < 6)
        return res.status(400).json({ msg: "Password must be at least 6 characters." });

    const passwordHash = await bcrypt.hash(password, 12);

    const newUser = new Users({
        username: newUserName, email, password: passwordHash
    });

    const access_token = createAccessToken({ id: newUser._id });
    const refresh_token = createRefreshToken({ id: newUser._id });

    res.cookie('refreshtoken', refresh_token, {
        httpOnly: true,
        path: '/api/refresh_token',
        maxAge: 30 * 7 * 24 * 60 * 60 * 1000 // 30days
    });

    await newUser.save();

    res.json({
        msg: 'Registered Successfully :)',
        access_token,
        user: {
            ...newUser._doc,
            password: ''
        }
    });
} catch (err) {
    return res.status(500).json({msg: err.message});
}
```

```
},
```

- **Add news function:**

Add news function is admin side functionality. Admin can select category, add title, description of news, images and add the news.

```
        addNews: async (req, res) => {
    try {
        const { title, images, content, category, user } = req.body;

        if (images.length === 0)
            return res.status(400).json({ msg: "Please add your photo." })

        if (!title)
            return res.status(400).json({ msg: "Please add title." });

        if (!content)
            return res.status(400).json({ msg: "Please add content." });

        if (!category)
            return res.status(400).json({ msg: "Please add Category." });

        const newNew = new News({
            title, images, content, category, user
        });
        await newNew.save();

        res.json({
            msg: 'News Added XD',
            newNew: {
                ...newNew._doc,
                user: req.user
            }
        })
    } catch (err) {
        return res.status(500).json({ msg: err.message });
    }
},
```

- **Update news function:**

  Update function is admin side functionality. Admin can modify news details.

```
editNews: async (req, res) => {
    try {
        const { title, images, content, category } = req.body;
        const news = await News.findByIdAndUpdate({_id: req.params.id}, {
            title, images, content, category
        });

        if (images.length === 0)
            return res.status(400).json({ msg: "Please add your photo." })

        if (!title)
            return res.status(400).json({ msg: "Please add title." });

        if (!content)
            return res.status(400).json({ msg: "Please add content." });

        if (!category)
            return res.status(400).json({ msg: "Please add Category." });

        res.json({
            msg: 'News Updated',
            newNew: {
                ...news._doc,
                user: req.user
            }
        })
    } catch (err) {
        return res.status(500).json({ msg: err.message });
    }
},
```

- **Delete news function:**
  Delete function is admin side functionality. Admin can Delete any news of his choice.

```
deleteNews: async (req, res) => {
```

```
    try {
        const news = await News.findByIdAndDelete({_id: req.params.id});

        res.json({
            msg: 'News Removed',
            news: {
                ...news,
                user: req.user
            }
        })
    } catch (err) {
        return res.status(500).json({ msg: err.message });
    }
},
```

- **Search news function:**

  Search function is user side functionality. User can search news by category of news or user can search news by headline of the news.

```
search: async (req, res) => {
    try {
        const news = await News.find({title: {$regex: req.query.news}});
        res.json({news});
    } catch (err) {
        return res.status(500).json({msg: err.message});
    }
},
```

- **Save -Unsave news function:**

  Users can save news of their choice by clicking on the save button and again click to unsave news.

```
saveNews: async (req, res) => {
    try {
        const user = await Users.find({_id: req.user._id, saved: req.params.id});

        if(user.length > 0)
            return res.status(400).json({msg: "You saved this post."});

        const save = await Users.findOneAndUpdate({_id: req.user.id}, {
            $push: {saved: req.params.id}
        }, {new: true})
```

```
            if(!save)
                return res.status(400).json({msg: 'This user does not exist.'})

            res.json({msg: 'News Saved'});
        } catch (err) {
            return res.status(500).json({msg: err.message});
        }
    },
    unSaveNews: async (req, res) => {
        try {
            const save = await Users.findOneAndUpdate({_id: req.params.user_id}, {
                $pull: {saved: req.params.id}
            }, {new: true});

            if(!save)
                return res.status(400).json({msg: 'This user does not exist.'})

            res.json({msg: 'unSaved news!'})
        } catch (err) {
            return res.status(500).json({msg: err.message});
        }
    },
    getSavedNews: async (req, res) => {
        try {
            const user = await Users.findById(req.user.id).select('-password');
            const saveNews = await News.find({
                _id: {$in: user.saved}
            });
            res.json({
                saveNews
            })
        } catch (err) {
            return res.status(500).json({msg: err.message});
        }
    },
```

- **Comment function:**
  Users can comment on the news and also they can view comments post by other users.

```
addcomment: async (req, res) => {
    try {
        const { content, postId, postUserId } = req.body;
```

```
        const news = await News.findById(postId);
        if(!news)
            return res.status(400).json({msg: "This News does not exist."});

        const newCmnt = new Comment({
            content, postId, postUserId, user: req.user.id
        });

        await News.findOneAndUpdate({_id: postId}, {
            $push: {comments: newCmnt._id}
        }, {new: true});

        await newCmnt.save();

        res.json({
            msg: 'News Added XD',
            newCmnt
        })
    } catch (err) {
        return res.status(500).json({ msg: err.message });
    }
},
comment: async (req, res) => {
    try {
        const comments = await Comment.find({postId:req.params.id});
        if(!comments)
            return res.status(400).json({msg: "This News does not exist"});

        res.json({
            comments
        })
    } catch (err) {
        return res.status(500).json({ msg: err.message });
    }
},
```

Testing:
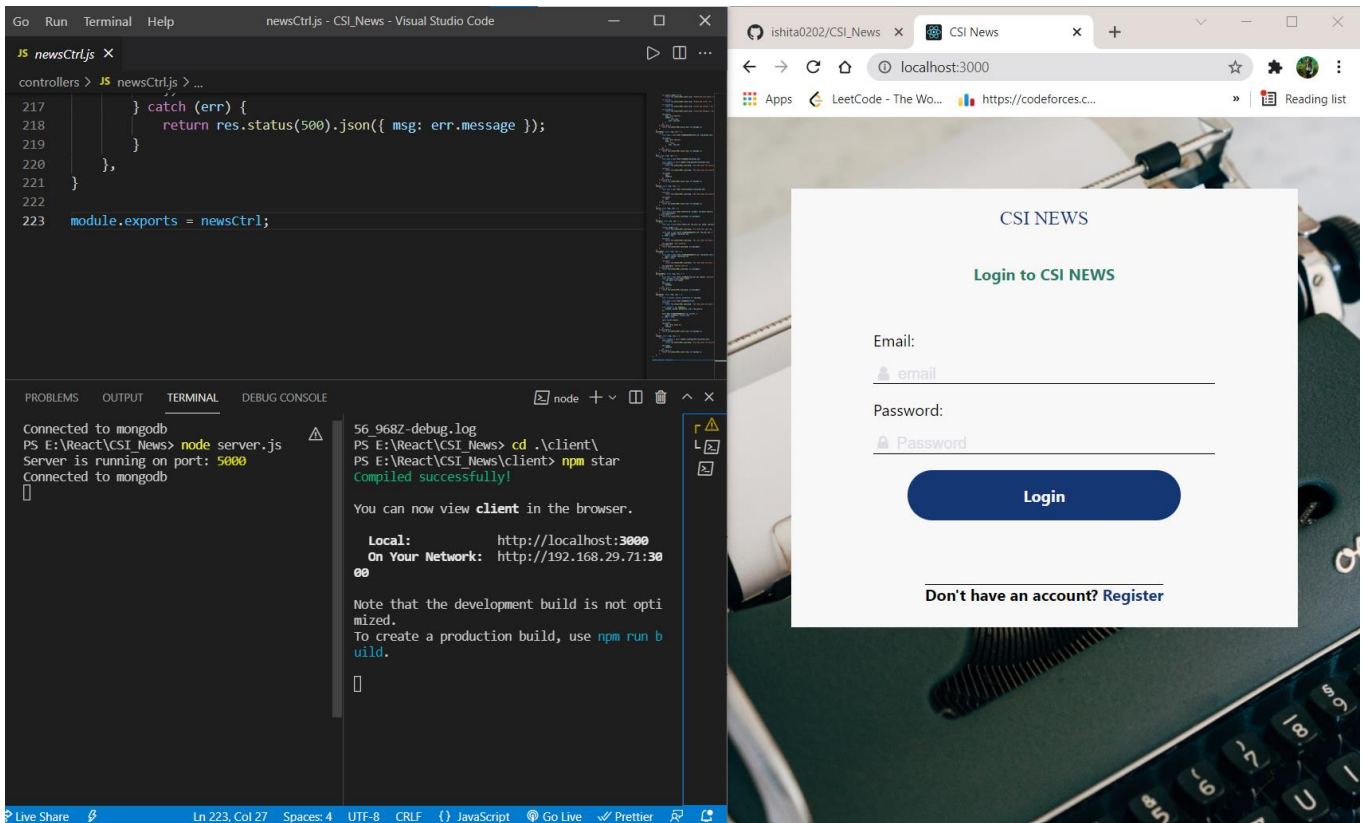
Test Cases

Test Case Id	SRS Id	Test Case Objective	Input Data	Expected Output	Actual Output	Status
TC_01	R 1.1	Admin login	Admin id, password	Success Message	Success Message	Pass

TC_02	R 1.2.2	Admin login	Wrong admin id	Admin not found	Admin not found	Pass
TC_03	R 1.2.3	Admin login	Wrong admin id	Admin not found	Admin not found	Pass
TC_04	R 2.2	User Login	User Id, Password	Success Message	Success Message	Pass
TC_05	R 2.2	User Login	Wrong User Email	User Not Found	User Not Found	Pass
TC_06	R 2.2	User Login	Wrong Password	Password is Wrong	Password is Wrong	Pass
TC_07	R 2.1.1	User Registration	Already user details	User already exist	User already exist	Pass
TC_08	R 2.1.1	User Registration	Valid user details	User Added	User Added	Pass
TC_09	R 2.1.3	Edit User details	Update Student Data	Updated Successfully	User updated	Pass
TC_10	R 3.1.1	Add news	Add news data	News added	News added	Pass
TC_11	R 3.3.2	Update news	Updated news	News Updated	News Updated	Pass
TC_12	R 3.3.3	Delete news	Delete	News deleted	News deleted	Pass

Screen-Shots:

Login
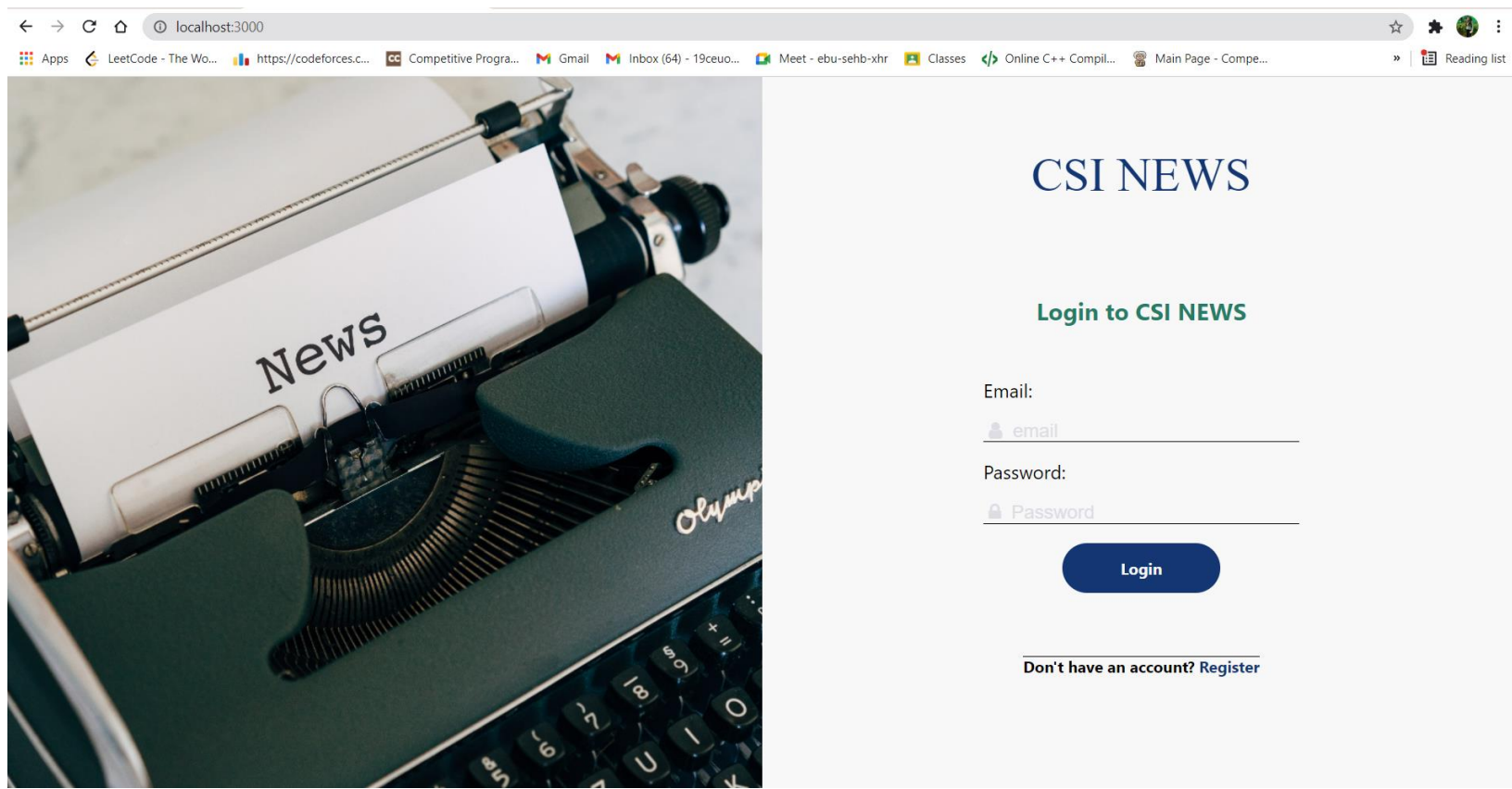
CSI NEWS

Login to CSI NEWS

Email:

👤 email

Password:

🔒 Password

Login

Don't have an account? Register

Registration

CSI NEWS

News   About US   Admin   Profile   Logout

Cars   Politics   Health   Sports   Education   Business   Science   World   Entertainment

Ancient

Online Banking Frauds: Here's how to protect yourself from phishing

Read More...

Ishaan Khatter visits Ananya Panday with bouquet of flowers after NCB quizzed

Read More...

Amit Shah in Jammu and Kashmir: Here's what's on agenda on day 2

Read More...

T20 World Cup: India aim to maintain all-win record against Pakistan

Read More...
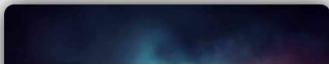
Ishaan Khatter visits Ananya Panday with bouquet of flowers after NCB quizzed her in Aryan Khan drugs case

Category: Entertainment
8 minutes ago

New Delhi: Ananya Panday's rumoured boyfriend Ishaan Khatter visited the actress on Saturday (October 23) with a bouquet of flowers at her residence. Ananya Panday has recently been going through a tough time as the Narcotics Control Bureau is questioning her in Aryan Khan's drugs case. The actress was summoned by the investigation agency on Friday and Saturday and was asked to appear again for questioning on Monday (October 25). Ananya, who appeared at the NCB office along with her father Chunkey Panday was grilled for 4 hours on Saturday. The actress was probed on her WhatsApp chats with Aryan Khan in which she allegedly arranged 'ganga' (weed) for the latter. "The chats recovered from Aryan Khan's mobile phone reveal that in 2018-19, she helped supply drugs to Aryan thrice by providing him the numbers of drug dealers," NCB sources told ANI. The actress has however denied all allegations of consuming and arranging drugs. Rumoured beau Ishaan, who worked with Ananya in the film Khaali Peeli, bought a huge bouquet of white, red and pink flowers, before paying a visit to Ananya's home. The two were earlier rumoured to have vacationed together in the Maldives. They have however stayed tight-lipped about the status of their relationship.

Comments:

ishita_0202: shocking!!!!

28

localhost:3000/news/6174ef490d2f3646ac2bee6f

Apps | LeetCode - The Wo... | https://codeforces.c... | Competitive Progra... | Gmail | Inbox (64) - 19ceuo... | Meet - ebu-sehb-xhr | Classes | Online C++ Compil... | Main Page - Compe... | » | Reading list

Ishaan Khatter visits Ananya Panday with bouquet of flowers after NCB quizzed her in Aryan Khan drugs case

Category: Entertainment
10 minutes ago

New Delhi: Ananya Panday's rumoured   ober 23) with a bouquet of flowers at her residence. Ananya Panday has recen   l Bureau is questioning her in Aryan Khan's drugs case. The actress was sum   nd was asked to appear again for questioning on Monday (October 25). A   Chunkey Panday was grilled for 4 hours on Saturday. The actress was prob   egedly arranged 'ganga' (weed) for the latter. "The chats recovered from Aryan   ly drugs to Aryan thrice by providing him the numbers of drug dealers," NCB   ons of consuming and arranging drugs. Rumoured beau Ishaan, who worked wi   white, red and pink flowers, before paying a visit to Ananya's home. The tw   Maldives. They have however stayed tight-lipped about the status of their rel

Add Comment

Content:

Enter content...

ADD

Comments:

UserProfile

Cars   Politics   Health   Sports   Education   Business   Science   World   Entertainment

Ancient

Saved News:

T20 World Cup: India...
18 minutes ago

Amit Shah in Jammu a...
15 minutes ago

Ishaan Khatter visit...
13 minutes ago

Online Banking Fraud...

ishita_0202
ishitaachauhan11@gmail.com
Create at a month ago

Edit Profile

Edit Profile

Apps   LeetCode - The Wo...   https://codeforces.c...   Competitive Progra...   Gmail   Inbox (64) - 19ceuo...   Meet - ebu-sehb-xhr   Classes   Online C++ Compil...   Main Page - Compe...   Reading list

CSI NEWS

Enter News name here

News   About US   Admin   Profile

Cars   Politics   Health   Science   World   Entertainment

Ancient

Change
Choose File   No file chosen

UserName:
ishita_0202

WebSite

UPDATE

ishita_0202

ishitaachauhan11@gmail.com

Create at a month ago

Edit Profile

Online Banking Fraud...

Serach news

31

CSI NEWS

ananya   ✕

News   About US   Admin   Profile   Logout

Cars   Politics   Health   Sports   Education   Business   Science   World   Entertainment

Ancient

Ishaan Khatter visits Ananya Panday with bouquet of flowers after NCB quizzed

Read More...

About Us and feedback

CSI NEWS

Enter News name here...

News   About US   Admin   Profile   Logout

Cars   Politics   Health   Sports   Education   Business   Science   World   Entertainment

Ancient

About Us:

Lorem ipsum dolor sit amet consectetur adipisicing elit. Possimus harum, laboriosam sapiente repudiandae veniam cumque vero eveniet doloremque maxime dolorum!

ishitaachauhan11@gmail.com

charvitardeshna@gmail.com

smitbhoraniya11@gmail.com

Contect Us:

Email Address:

Enter Email Address...

Message:

Enter Message...

FEEDBACK

Admin-side:

Add news

Edit news

Delete news

Conclusion:

Hence-forth in this project we have successfully implemented the admin-side & User-side functionality, they will find headlines, breaking news and news of category of their choice. Users can save the news and also, they can comment on news and view other users' comments. Users can modify their profile. Admin can login to the system. Admin can Add news, Update News and Delete news.

Limitations:

1) There is no filter add filter system news section.
2) User cannot download the news.
3) User cannot change the password after registration.

Future Extension:

To take over the limitations we are planning this future extension in our system.

4) We are planning to add filter system news section.
5) Download local newspaper.
6) Special category for daily news.

Bibliography

References/resources used for developing project:

1. https://reactjs.org/
2. https://stackoverflow.com/
3. https://timesofindia.indiatimes.com/
4. https://zeenews.india.com/
5. https://www.mongodb.com/mern-stack