**A PROJECT REPORT**

ON

Flash-Card App

(For Creating fast FlashCard(note) App)

By

**PATEL DHRUV (CE108)(19CEUBG166)**

**PARMAR HIMANSHU (CE103) (19CEUSG047)**

B.Tech CE Semester-V

Subject: Subject: Advance Technologies

Guided by: Prof. Siddharth P. Shah

Assistant Professor

Dept. of Comp. Eng.

Faculty of Technology Department of Computer Engineering

Dharmsinh Desai University

# CERTIFICATE

This is to certify that the practical / term work carried out in the subject of

**Advance Technologies** and recorded in this journal is the

bonafide work of

**PATEL DHRUV J. (CE108)(19CEUBG166)**

**PARMAR HIMANSHU V. (CE103) (19CEUSG047)**

of B.Tech semester V in the branch of Computer Engineering

during the academic year **2021-2022.**

Prof. Siddharth P. Shah
Assistant Professor,
Dept. of Computer Engg.,
Faculty of Technology
Dharmsinh Desai University, Nadiad

Dr. C. K. Bhensdadia,
Head,
Dept. of Computer Engg.,
Faculty of Technology
Dharmsinh Desai University,Nadiad

# Table of Content

# 1.Introduction

## 1.1Brief Introduction

- “Flash-card” is an online App. This allows to create flashcards very fast.so the user get good experience. In today’s time flash card is important. Online Flashcards can be a life saver! Flashcards are the perfect study tool for today’s students, especially now you can create Flashcards online with our Flash-card App.

- The benefits include improving language skills, increasing the ability to compose stories, memorizing, analyzing a problem, and enriching vocabulary. Apart from the cognitive side, the benefits of a flashcard can also increase self-confidence, develop good and effective communication, and enhance creativity.

- Flashcards allow you to study in a variety of different ways. Unlike a lot of other study resources, the benefits of using flashcards to study can be noticed straight away. Whether you’re doing last minute prep for your exam or just brushing up on your specific Subject(topic), online Flashcards are perfect for memorising key facts quickly.

- Flash-Card is a tool that helps you to study on the travel, you save the environment, you also helps to your friend by sharing flashcards and you also save yourself bunch of time.

## 1.2 Tools/Technologies Used

- Technologies:
  - React.js
  - Node.js
  - HTML
  - CSS
  - JavaScript
  - JSX
  - Express.js
  - MongoDB
  - Bcrypt

- Tools
  - Git
  - GitHub
  - Visual Studio Code

- Platform
  - Local development server
  - MongoDB Compass

# 2. Software Requirement Specifications

## 2.1System Functional Requirements

### R1 Authentication

R1.1 Registration

Description: If user is new, it gets register himself with email address and password.

Input: Valid e-mail address and strong password.

Output: Account created and redirected to Login.

R1.2 Login

Description: User should login through registered e-mail id and password.

Input: E-mail id and password

Output: Successfully login redirected to Home.

R1.3 Logout

Description: User can logout using logout button.

Input: no input require. click on logout button

Output: Successfully Logout

**R2 CRUD  Operation on flashcard**

R2.1 Add Flash-Card

Description: User should create flash card to provide information like question, answer & tag.

Input: Question, Answer & Tag

Output: Successfully Added flash card and listed.

R2.2 Update Flash-Card

Description: User should update flash card using edit icon.

Input: Question, Answer & Tag

Output: Successfully Updated flash card and listed.

R2.3 Delete Flash-Card

Description: User should delete flash card using delete icon.

Input: no input require. click on delete icon

Output: Successfully Deleted flash card.

R2.4 Read Flash-Card

Description: User should read flash card on that page.

Input: no input require.

Output: no output

R3 Change to Dark Mode

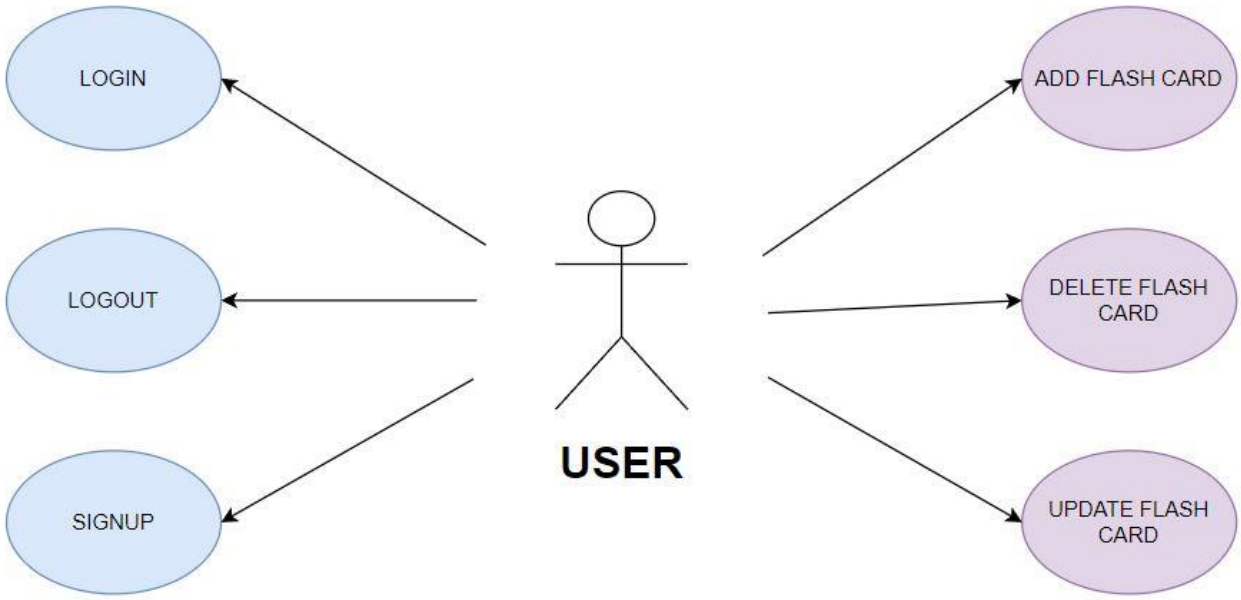
Description: User should change UI using dark mode enable.

Input: click on dark mode switch

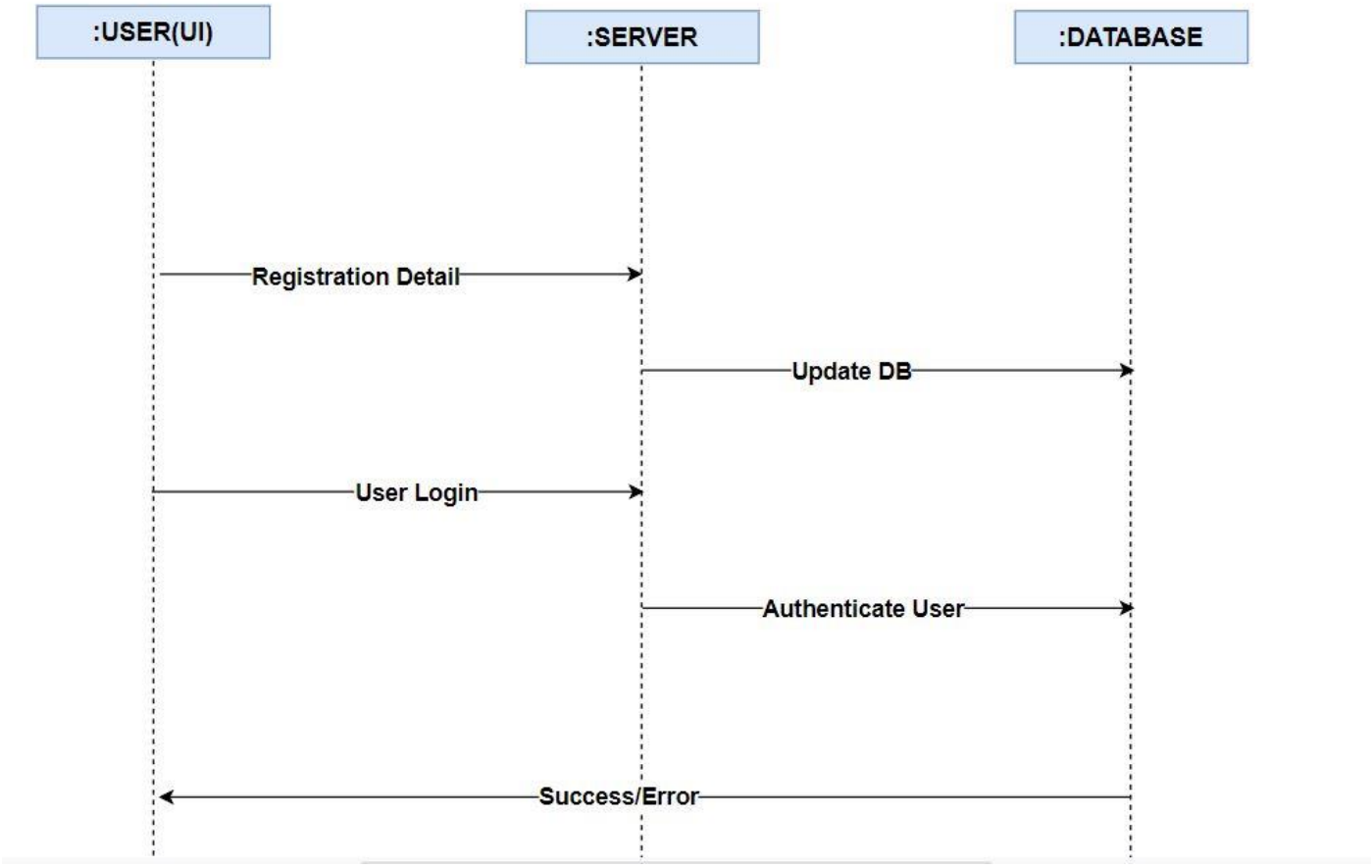
Output: Dark mode is on

# 3. Design

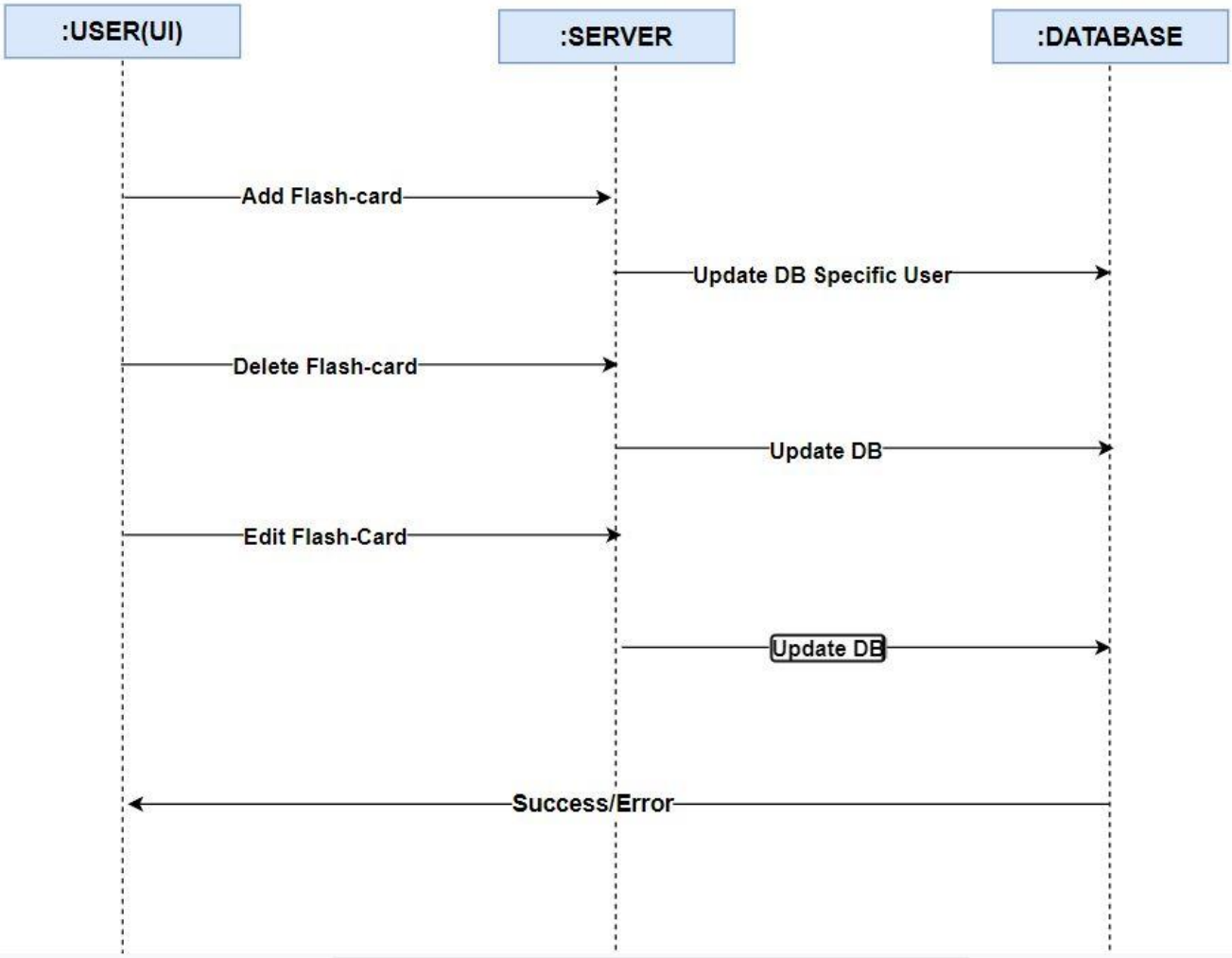
## 3.1Use Case Diagram



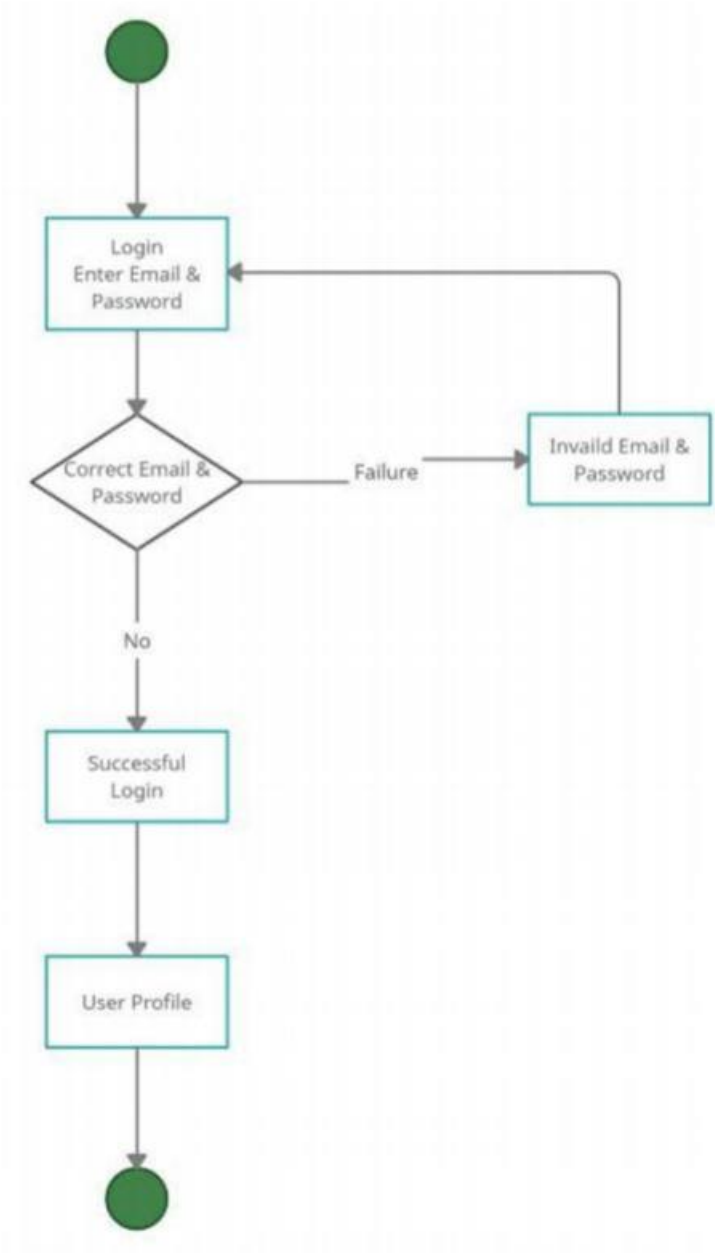
## 3.2Sequence Diagram(User Authentication)



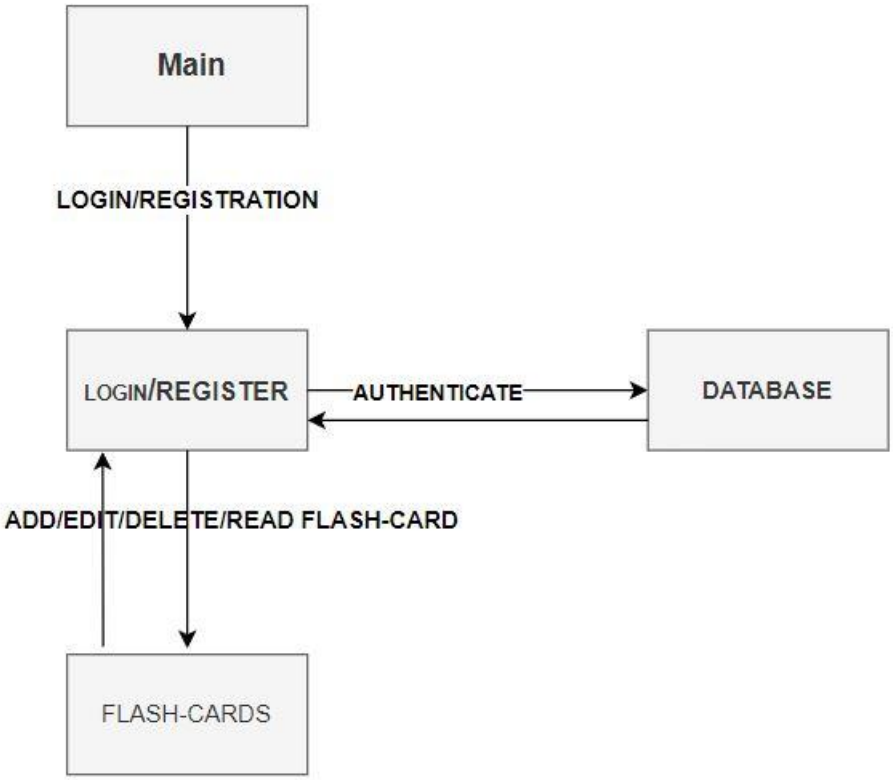
## 3.2Sequence Diagram(Flash Card CRUD)



## 3.3ActivityDiagram

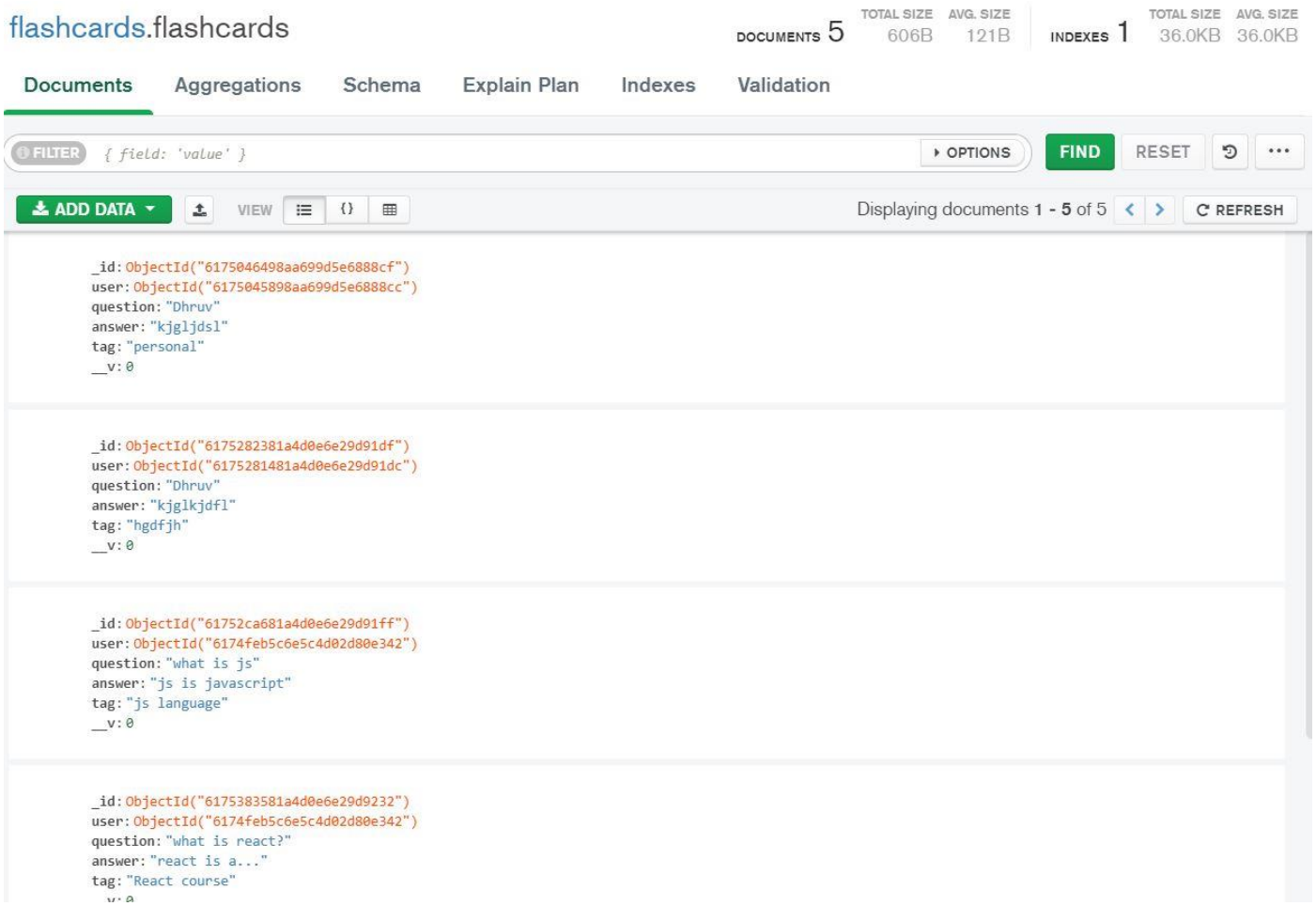


### 3.4 Structure Chart



### 3.5 Data Dictionary

In our project we have used the NoSQL 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 MongoDB Compass. It doesn't have a table, row & column but it has a collection document & field as shown in below pictures.



flashcards.users

DOCUMENTS 3 TOTAL SIZE 491B AVG. SIZE 164B | INDEXES 1 TOTAL SIZE 36.0KB AVG. SIZE 36.0KB

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' } OPTIONS FIND RESET

ADD DATA VIEW Displaying documents 1 - 3 of 3 REFRESH


```
_id: ObjectId("6174feb5c6e5c4d02d80e342")
name: "Dhruv"
password: "$2a$10$BITRcLWt6UC5F.pKaTp3rudeKKV1EfL8bECIV1r8IOZNvXNlIkqZW"
email: "Dhruv@gmail.com"
date: 2021-10-24T06:35:33.350+00:00
__v: 0

_id: ObjectId("6175045898aa699d5e6888cc")
name: "sanket"
password: "$2a$10$s5TWZ81R/Ufu2awGTjFMN.MtpPgGBPZ.1g/bZLml9I/dZQ4gspPMO"
email: "sanket@gmail.com"
date: 2021-10-24T06:59:36.778+00:00
__v: 0

_id: ObjectId("6175281481a4d0e6e29d91dc")
name: "vidhi"
password: "$2a$10$OHbnvAP.KkpFklixzS/gw.RqZG/Uq8PAdp8p5Nw.i5vE.eDdO.VGu"
email: "vidhi@gmail.com"
date: 2021-10-24T09:32:04.732+00:00
__v: 0
```


Collections

CREATE COLLECTION

| Collection Name | Documents | Avg. Document Size | Total Document Size | Num. Indexes | Total Index Size | Properties |
|---|---|---|---|---|---|---|
| flashcards | 5 | 121.2 B | 606.0 B | 1 | 36.0 KB | |
| users | 3 | 163.7 B | 491.0 B | 1 | 36.0 KB | |

# 4. Implementation Details

## 4.1 Function prototypes

```
//  Route:1-->create a new user

router.post("/createuser", [body('name').isLength({ min: 3 }), body('password').isLength({ min: 5 }), body('email').isEmail()],
    async(req, res) => {
        const error = validationResult(req);
        if (!error.isEmpty()) {
            return response.status(400).json({
                erro: error.message
            });
        }
        try {
            let user = await User.findOne({ email: req.body.email });
            if (user) {
                return res.status(400).json({ error: "Enter Valid Credential." })
            }
            const salt = await bcrypt.genSalt(10);
            const secPass = await bcrypt.hash(req.body.password, salt);
            user = await User.create({ name: req.body.name, password: secPass, email: req.body.email });

            const data = {
                user: {
                    id: user.id
                }
            }
            const authtoken = jwt.sign(data, JWT_SECRET)
            let success=true;
            res.json({
                success,authtoken
            })
        } catch (error) {
            console.error(error.message);
            res.status(500).send("Server Not Responded");
        }
    }
)
```

**Create New User**

```
//  Route:2-->User Login

router.post("/login", [body('password').isLength({ min: 5 }), body('email').isEmail()],
    async(req, res) => {
        const error = validationResult(req);
        if (!error.isEmpty()) {
            return response.status(400).json({
                erro: error.message
            });
        }
        try {
            const { email, password } = req.body;
            let user = await User.findOne({ email });
            if (!user) {
                return res.status(400).json({ error: "Enter Correct Credential." })
            }
            const passwordcompare = await bcrypt.compare(password, user.password)
            if (!passwordcompare) {
                return res.status(400).json({ error: "Enter Correct Credential." })
            }

            const data = {
                user: {
                    id: user.id
                }
            }
            const authtoken = jwt.sign(data, JWT_SECRET)
            let success=true;
            res.json({
                success,authtoken
            })
        } catch (error) {
            console.error(error.message);
            res.status(500).send("Server Not Responded");
        }
    }
)
```

**User Login**

```
//  Route:3-->User Details

router.post("/getuser", fetchuser, async(req, res) => {
    try {
        userId = req.user.id;
        const user = await User.findById(userId).select("-password");
        let success=true;
        res.send(success,user);
    } catch (error) {
        console.error(error.message);
        res.status(500).send("Server Not Responded");
    }
})
```

**Fetch User Details**

```
//  Route:1--> Get all the flashcards for a particular user

router.get("/getcards", fetchuser, async(req, res) => {
    try {
        const cards = await Flashcards.find({ user: req.user.id });
        res.json(cards);
    } catch (error) {
        console.error(error.message);
        res.status(500).send("Server Not Responded");
    }
})
```

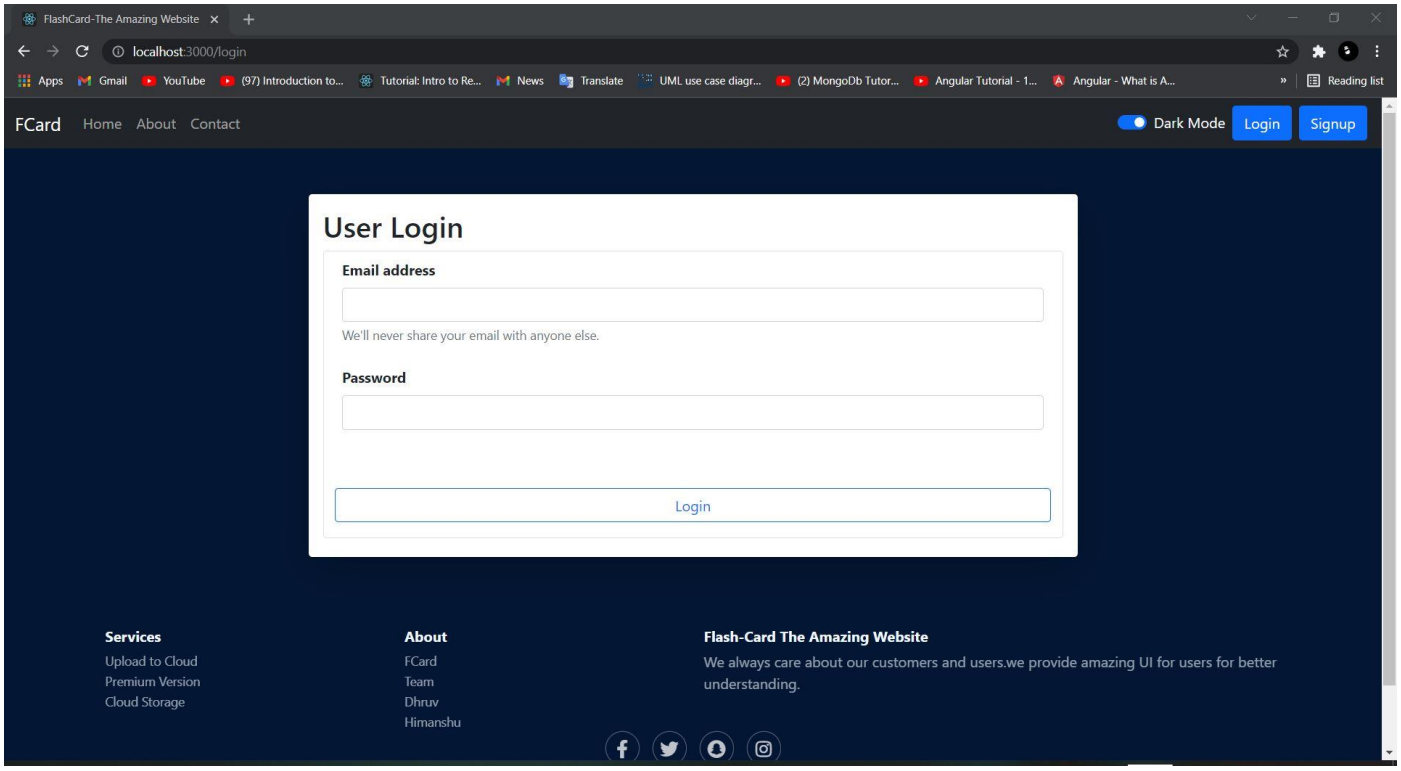
```
//  Route:2--> Add new flashcard for a particular user

router.post("/addcard", fetchuser, [body('question').isLength({ min: 5 }), body('answer').isLength({ min: 5 }), body('tag').isLength({ min: 2 })],
async(req, res) => {

    const error = validationResult(req);
    if (!error.isEmpty()) {
        return response.status(400).json({
            erro: error.message
        });
    }
    try {
        const { question, answer, tag } = req.body;
        const newcard = new Flashcards({ question, answer, tag, user: req.user.id });
        const savedcard = await newcard.save()
        res.json(savedcard);
    } catch (error) {
        console.error(error.message);
        res.status(500).send("Server Not Responded");
    }
})
```

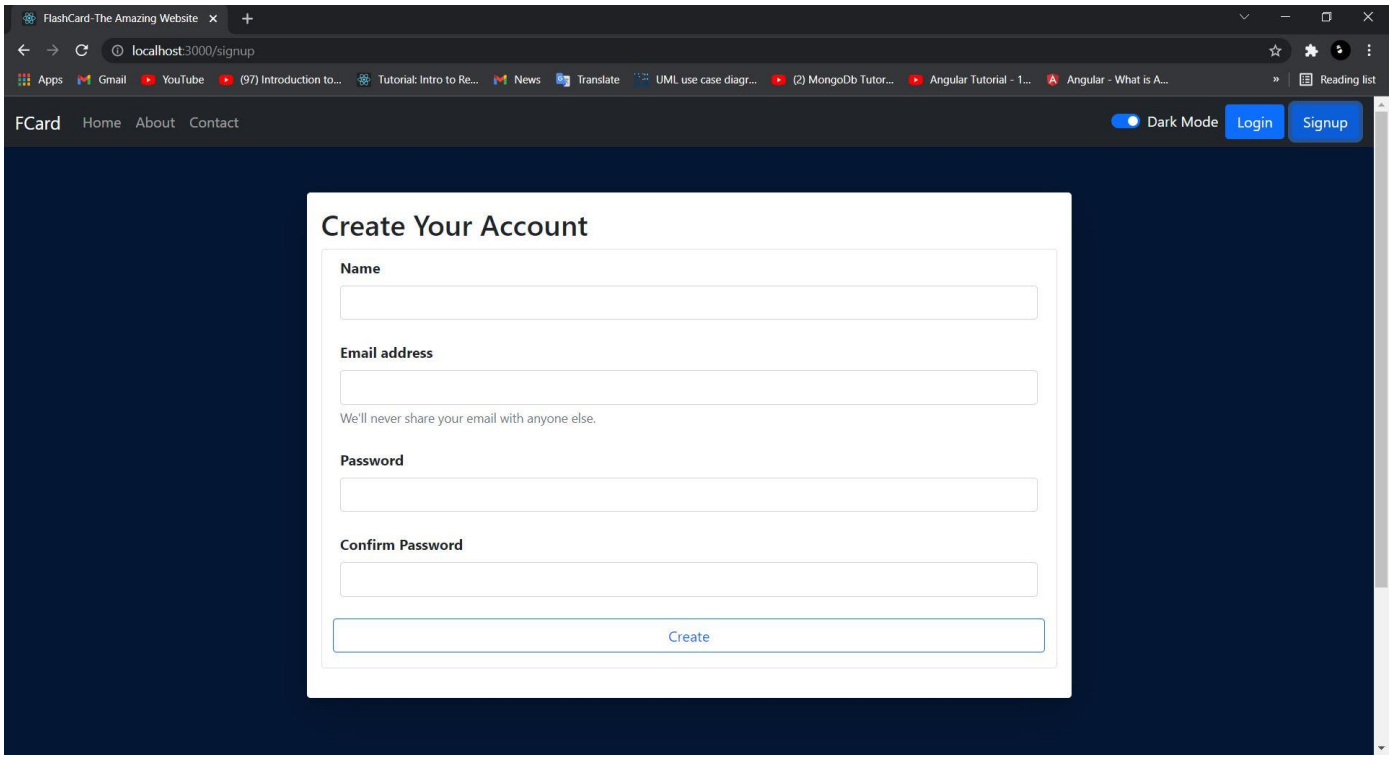
```
//  Route:4--> Delete flashcard for a particular user

router.delete("/deletecard/:id", fetchuser, async(req, res) => {
    try {

        let card = await Flashcards.findById(req.params.id);
        if (!card) {
            return res.status(404).send("Not Found!!");
        }
        if (card.user.toString() !== req.user.id) {
            return res.status(401).send("Not Allowed!!");
        }
        card = await Flashcards.findByIdAndDelete(req.params.id);
        res.json({ "success": "Card has been deleted successfully!!", card: card });
    } catch (error) {
        console.error(error.message);
        res.status(500).send("Server Not Responded!!");
    }
})
```

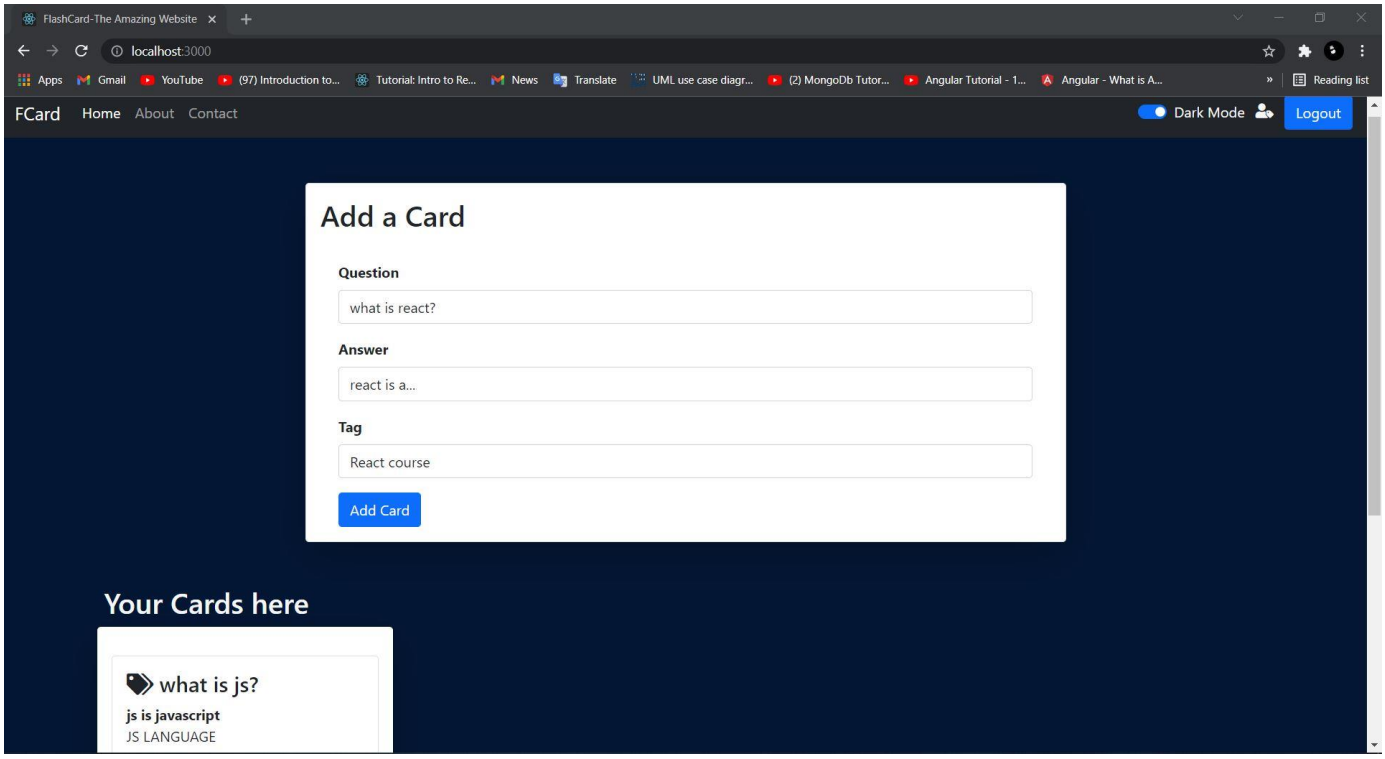
```
//  Route:3--> Update flashcard for a particular user

router.put("/updatecard/:id", fetchuser, [body('question').isLength({ min: 5 }), body('answer').isLength({ min: 5 }), body('tag').isLength({ min:
2 })], async(req, res) => {
    const error = validationResult(req);
    if (!error.isEmpty()) {
        return response.status(400).json({
            erro: error.message
        });
    }
    try {
        const { question, answer, tag } = req.body;
        const updatedcard = {};
        if (question) { updatedcard.question = question }
        if (answer) { updatedcard.answer = answer }
        if (tag) { updatedcard.tag = tag }

        let card = await Flashcards.findById(req.params.id);
        if (!card) {
            return res.status(404).send("Not Found!!");
        }
        if (card.user.toString() !== req.user.id) {
            return res.status(401).send("Not Allowed!!");
        }
        card = await Flashcards.findByIdAndUpdate(req.params.id, { $set: updatedcard }, { new: true });
        res.json({ updatedcard });
    } catch (error) {
        console.error(error.message);
        res.status(500).send("Server Not Responded!!");
    }
})
```

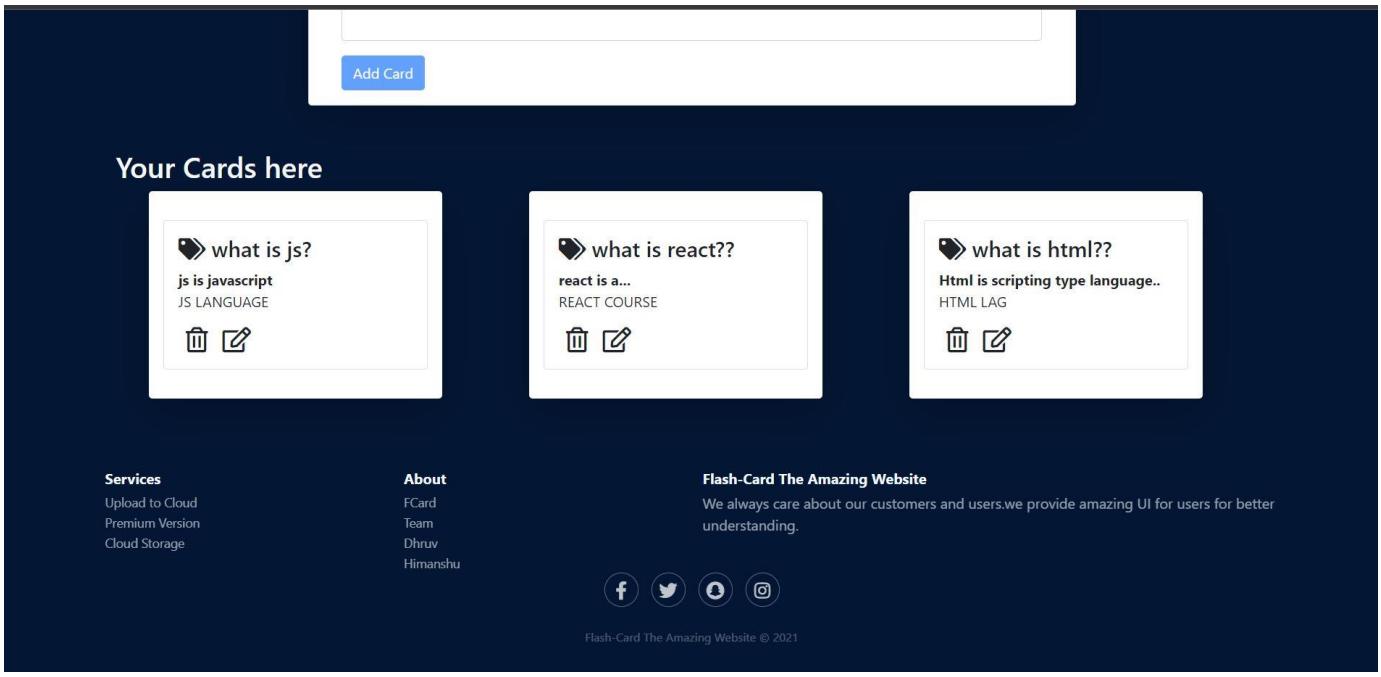
## Flash-Card CRUD Operations

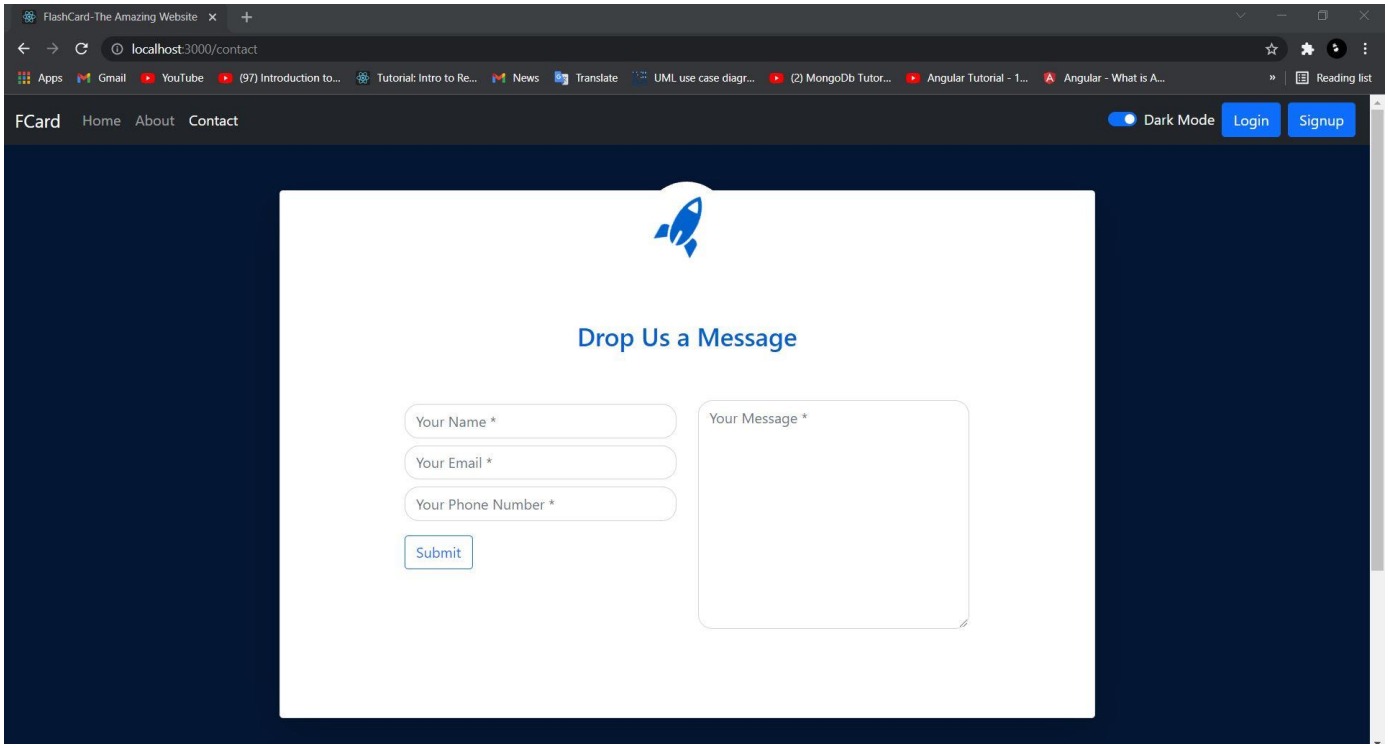
## <u>5 Screenshots</u>

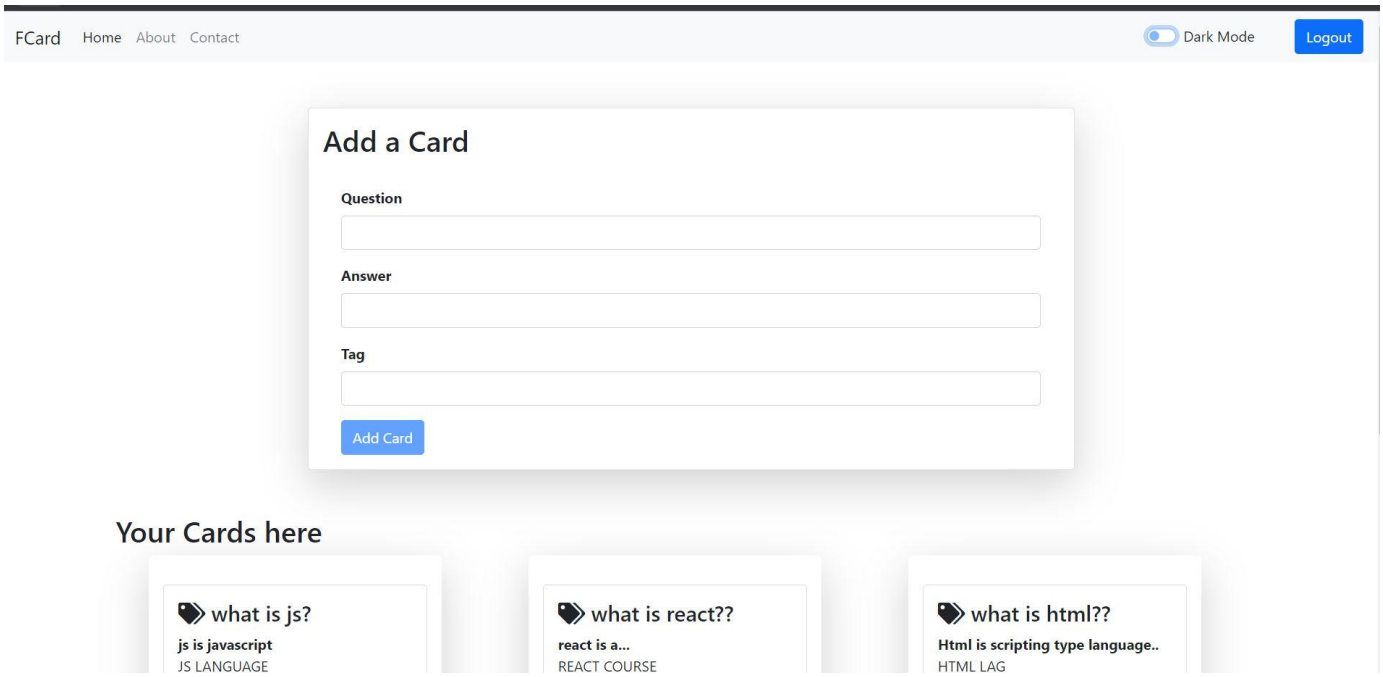



FlashCard-The Amazing Website
localhost:3000/signup
FCard
Home
About
Contact
Dark Mode
Login
Signup
Create Your Account
Name
Email address
We'll never share your email with anyone else.
Password
Confirm Password
Create


FlashCard-The Amazing Website
localhost:3000
FCard
Home
About
Contact
Dark Mode
Logout
Add a Card
Question
what is react?
Answer
react is a...
Tag
React course
Add Card
Your Cards here
what is js?
js is javascript
JS LANGUAGE


Add Card
Your Cards here
what is js?
js is javascript
JS LANGUAGE
what is react??
react is a...
REACT COURSE
what is html??
Html is scripting type language..
HTML LAG
Services
Upload to Cloud
Premium Version
Cloud Storage
About
FCard
Team
Dhruv
Himanshu
Flash-Card The Amazing Website
We always care about our customers and users.we provide amazing UI for users for better understanding.
Flash-Card The Amazing Website © 2021


FlashCard-The Amazing Website
localhost:3000/contact
FCard
Home
About
Contact
Dark Mode
Login
Signup
Drop Us a Message
Your Name *
Your Email *
Your Phone Number *
Submit
Your Message *




# 6 Limitations

- In our app search functionality is not added. if user search some specific flashcard using question or tag then app doesn't have that functionality.
- 2-step verification for more security is also not added.
- GUI is also not very much good.
- specific subject wise flashcards functionality is also not added.

## 6.1Future Enhancements

- In future we will add all the functionality mention above.
- In future we can make GUI much better and fast ,we can add

  Google login or Facebook login or GitHub login, and remove some bugs

  mention above.

## 7 Reference / Bibliography

Following links and websites were referred during the development of this project:

https://stackoverflow.com/

https://getbootstrap.com/docs/5.0/

https://reactjs.org/

https://expressjs.com/

https://nodejs.org/en/docs/

https://www.npmjs.com/package/bcrypt/

https://github.com/

https://mongoosejs.com/docs/

https://docs.mongodb.com/