Department of Information Technology, PICT Pune

Third Year Information Technology (2019 Course)314458:

Laboratory Practice-II (Cloud Computing)

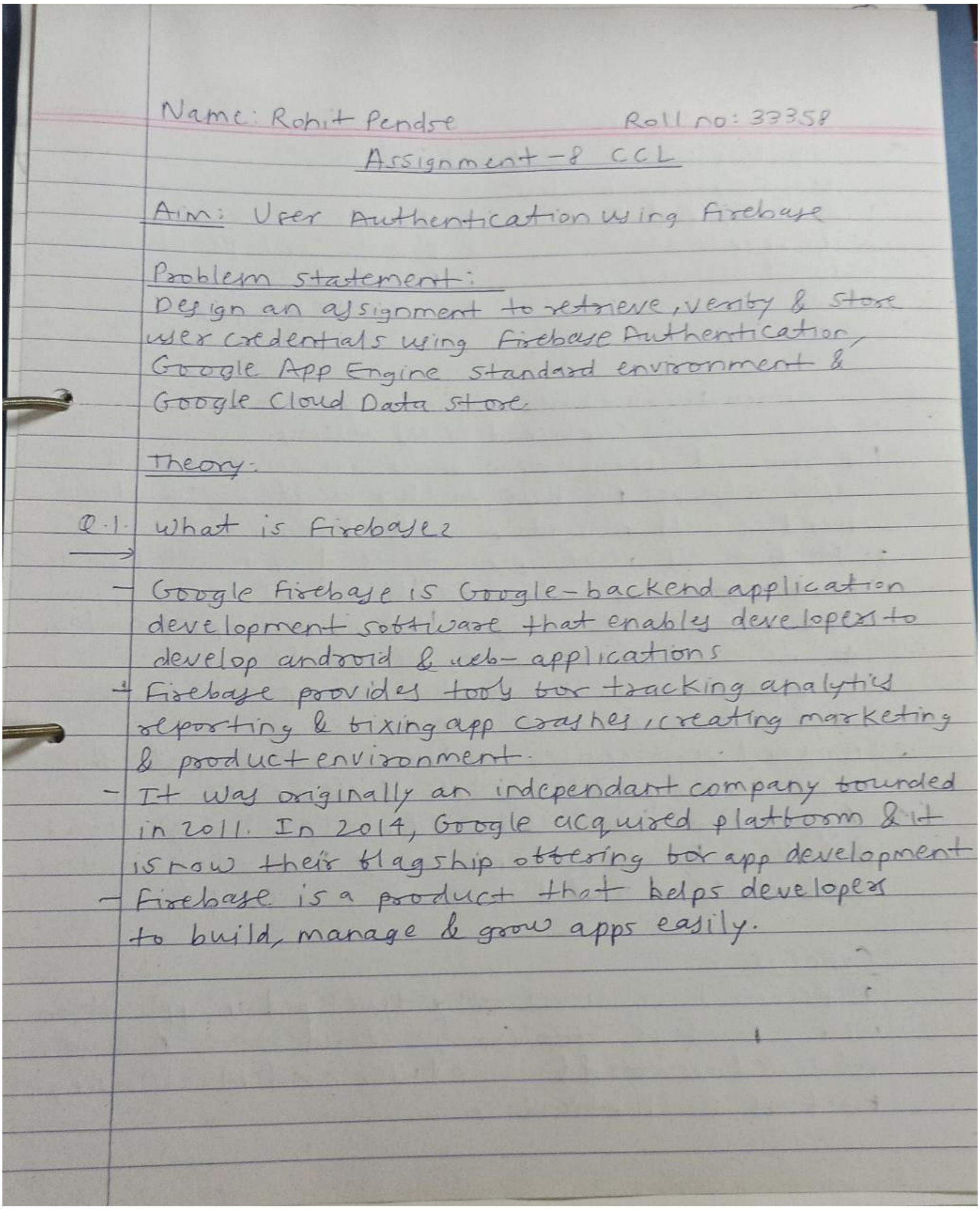
Assignment No. : 08

Design and develop “Contact Form” custom application using Firebase Authentication and the Google App Engine.

Name and roll number of group members:

Rohit Pendse	**33358**
Harshal Rajput	**33363**
Shubham Saraf	**33369**
Atharva Sarwate	**33370**

Guided by Prof. Sachin Pande



Name: Rohit Pendse Roll no: 33358

Assignment-8 CCL

Aim: User Authentication using firebase

Problem statement:

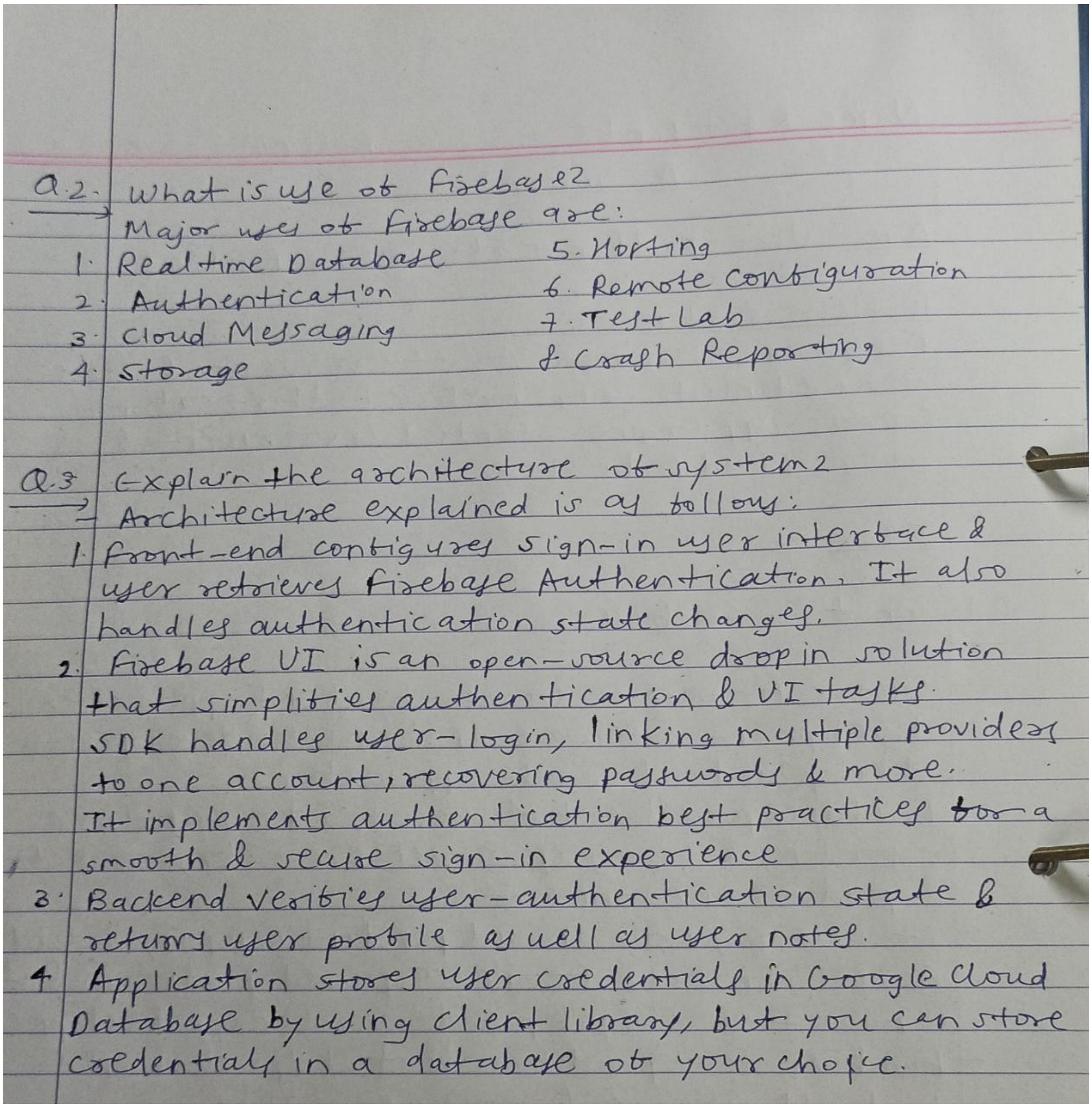
Design an assignment to retrieve, verify & store user credentials using Firebase Authentication, Google App Engine standard environment & Google Cloud Data store.

Theory:

Q.1. What is firebase?

→

- Google firebase is Google-backend application development software that enables developers to develop android & web-applications
- Firebase provides tools for tracking analytics reporting & fixing app crashes, creating marketing & product environment.
- It was originally an independant company founded in 2011. In 2014, Google acquired platform & it is now their flagship offering for app development
- Firebase is a product that helps developers to build, manage & grow apps easily.



Q.2. What is use of Firebase?

→ Major uses of Firebase are:

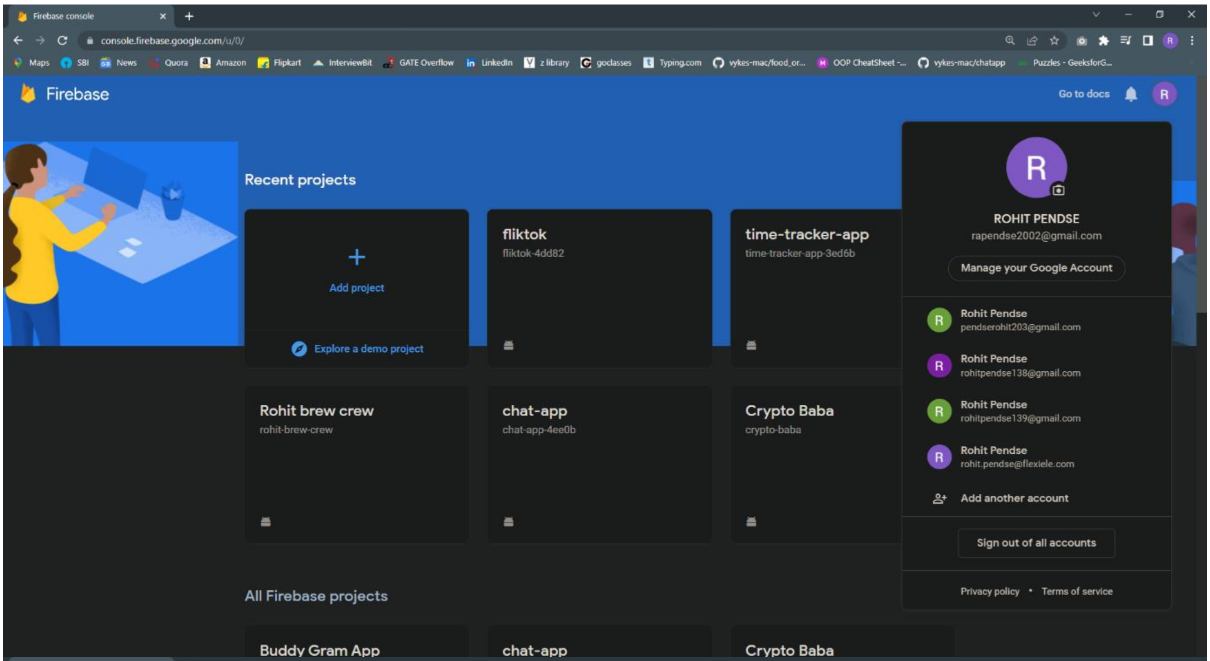
1. Realtime Database
2. Authentication
3. Cloud Messaging
4. Storage
5. Hosting
6. Remote Configuration
7. Test Lab
8. Crash Reporting

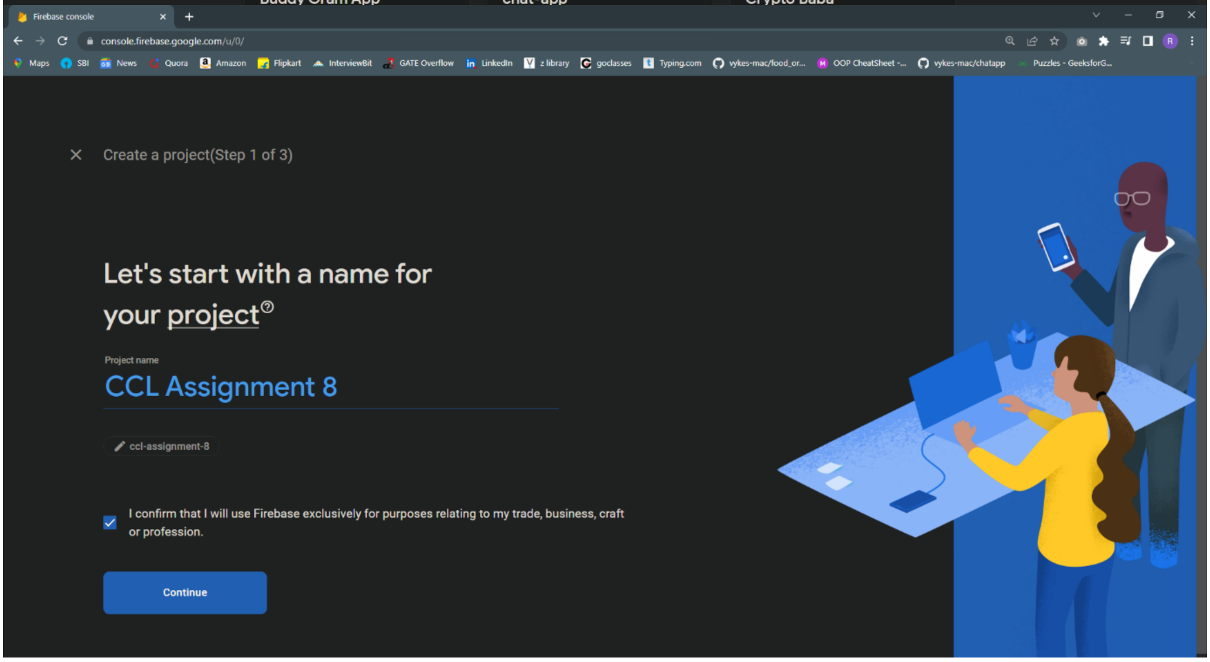
Q.3 Explain the architecture of system?

→ Architecture explained is as follows:

1. Front-end configures sign-in user interface & user retrieves Firebase Authentication. It also handles authentication state changes.
2. Firebase UI is an open-source drop in solution that simplifies authentication & UI tasks. SDK handles user-login, linking multiple providers to one account, recovering passwords & more. It implements authentication best practices ~~for~~ a smooth & secure sign-in experience
3. Backend verifies user-authentication state & returns user profile as well as user notes.
4. Application stores user credentials in Google Cloud Database by using client library, but you can store credentials in a database of your choice.

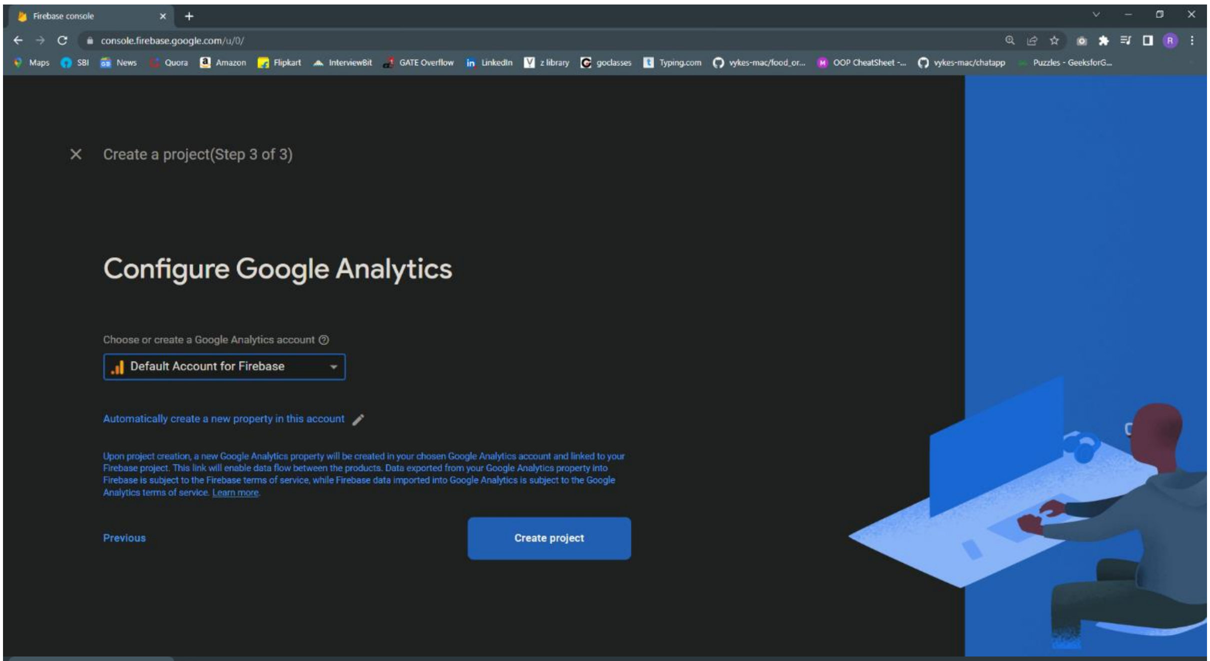
1. Initial Setup:




Firebase console
console.firebase.google.com/u/0/
Create a project(Step 1 of 3)
Let's start with a name for your project
Continue

Project name
CCL Assignment 8
ccl-assignment-8

☑ I confirm that I will use Firebase exclusively for purposes relating to my trade, business, craft or profession.



Create a project(Step 3 of 3)

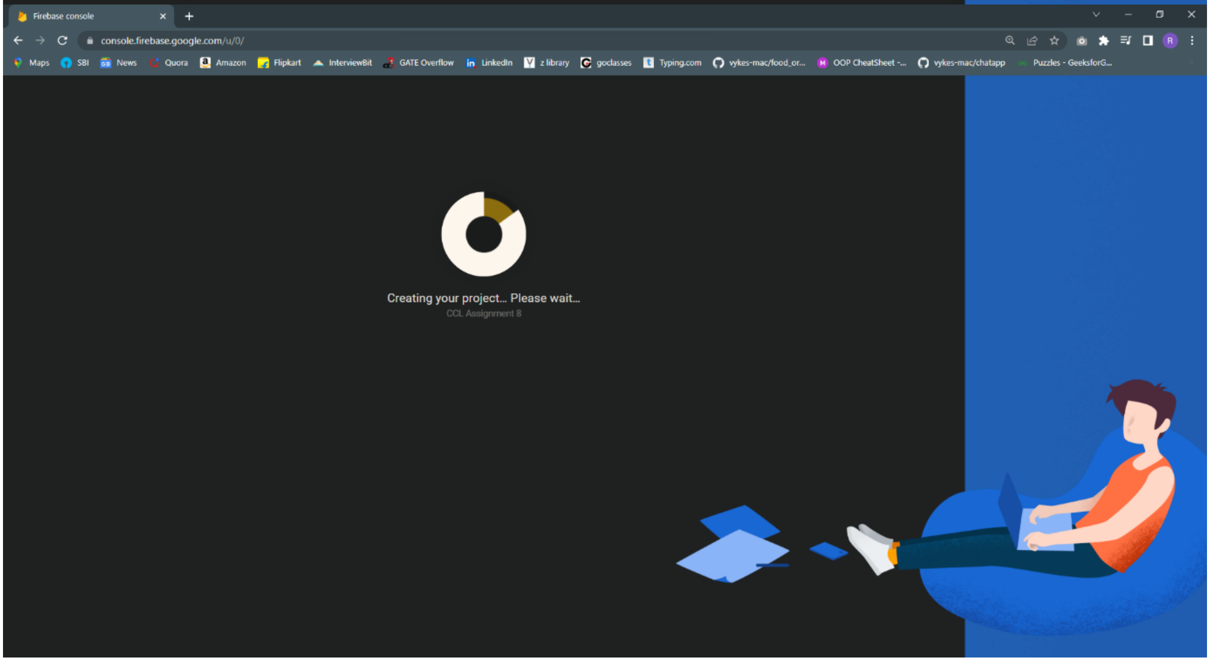
Configure Google Analytics

Choose or create a Google Analytics account

Default Account for Firebase

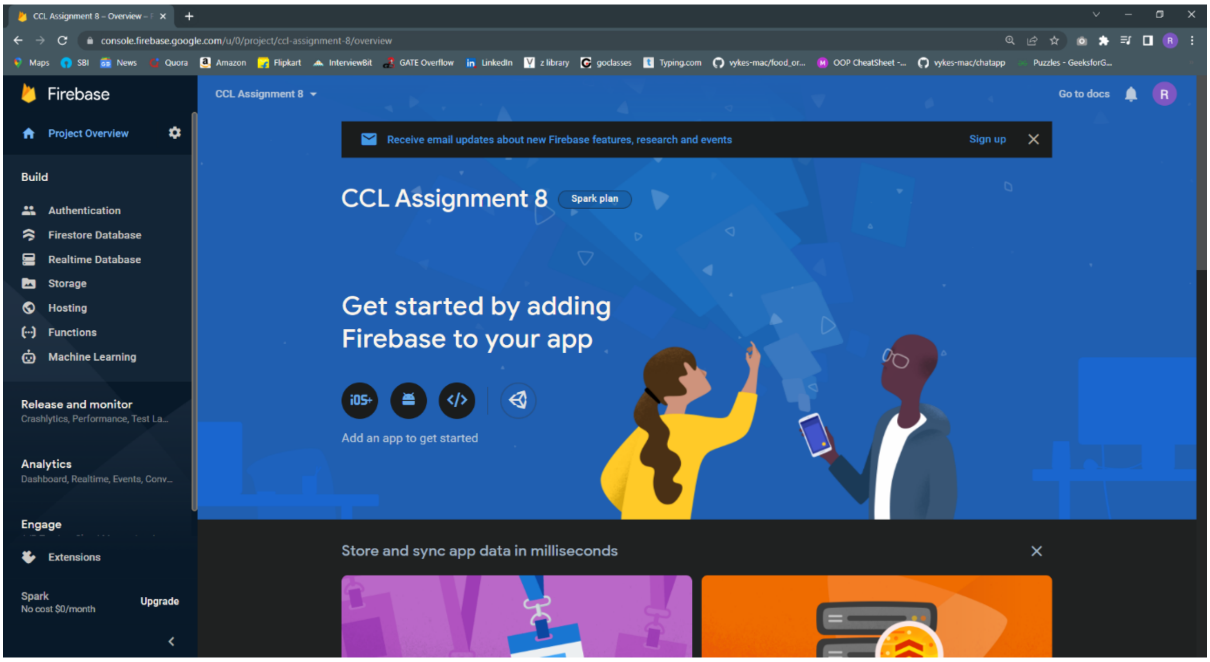
Automatically create a new property in this account

Upon project creation, a new Google Analytics property will be created in your chosen Google Analytics account and linked to your Firebase project. This link will enable data flow between the products. Data exported from your Google Analytics property into Firebase is subject to the Firebase terms of service, while Firebase data imported into Google Analytics is subject to the Google Analytics terms of service. Learn more.



Creating your project... Please wait...

CCL Assignment 8



Receive email updates about new Firebase features, research and events

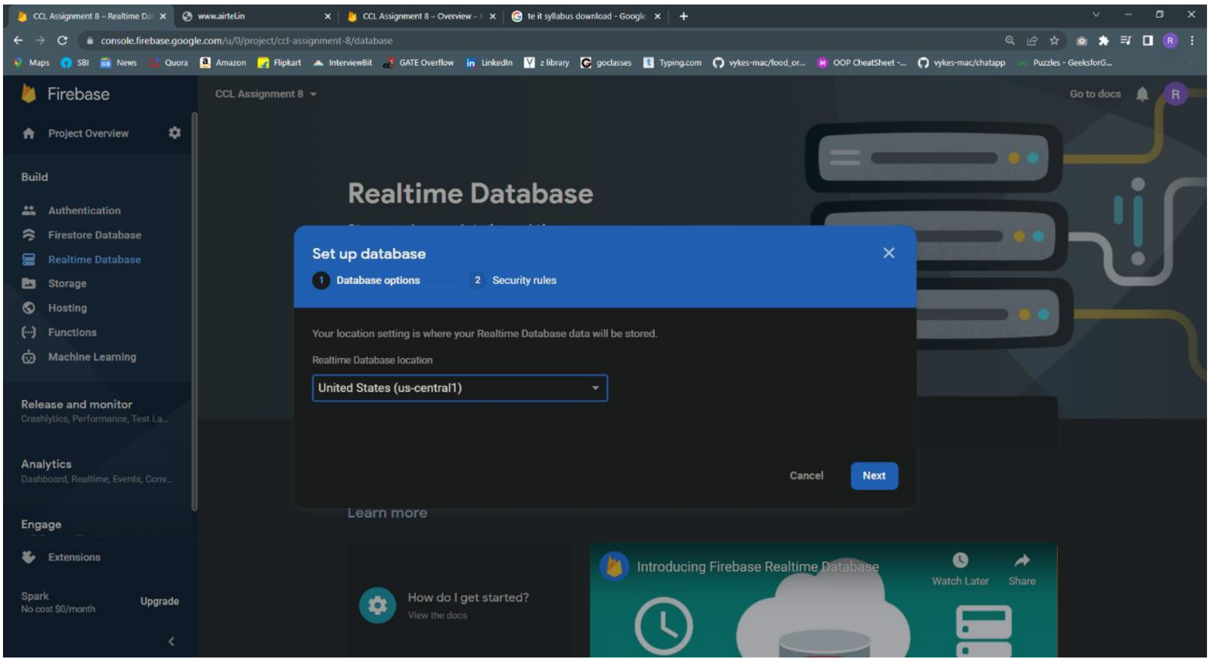
CCL Assignment 8

Get started by adding Firebase to your app

Add an app to get started

Store and sync app data in milliseconds

2. Setting up Realtime Database:



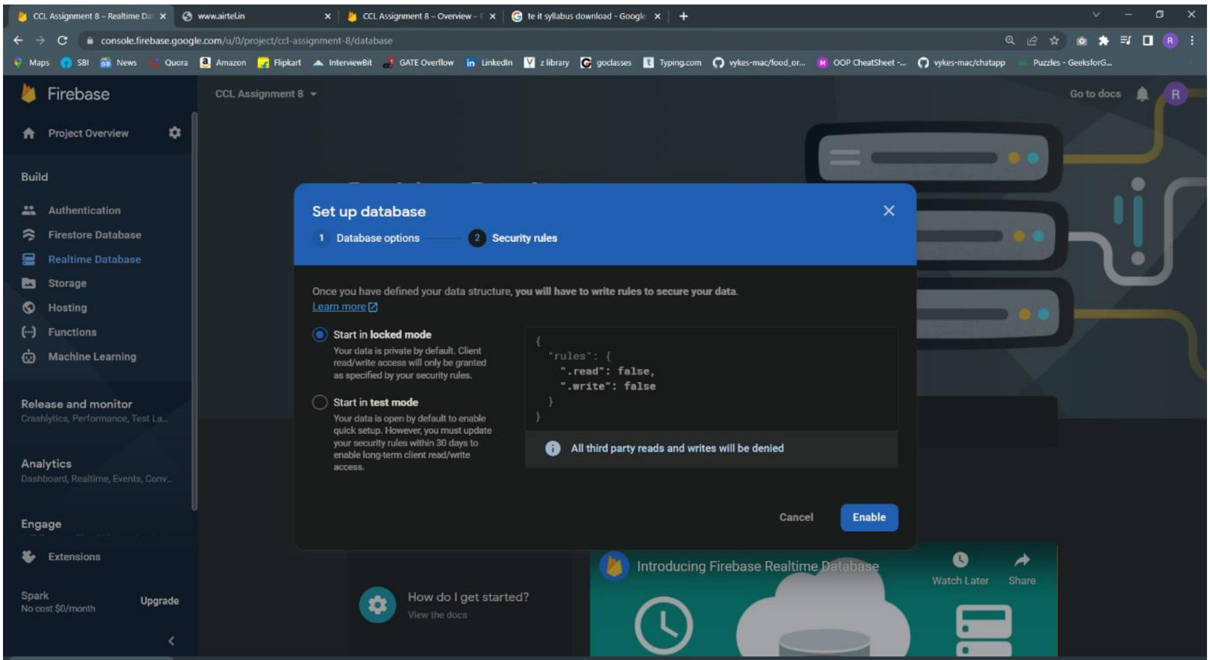
Set up database

1 Database options 2 Security rules

Your location setting is where your Realtime Database data will be stored.

Realtime Database location: United States (us-central1)

Cancel Next



Set up database

1 Database options — 2 Security rules

Once you have defined your data structure, **you will have to write rules to secure your data.**

Learn more

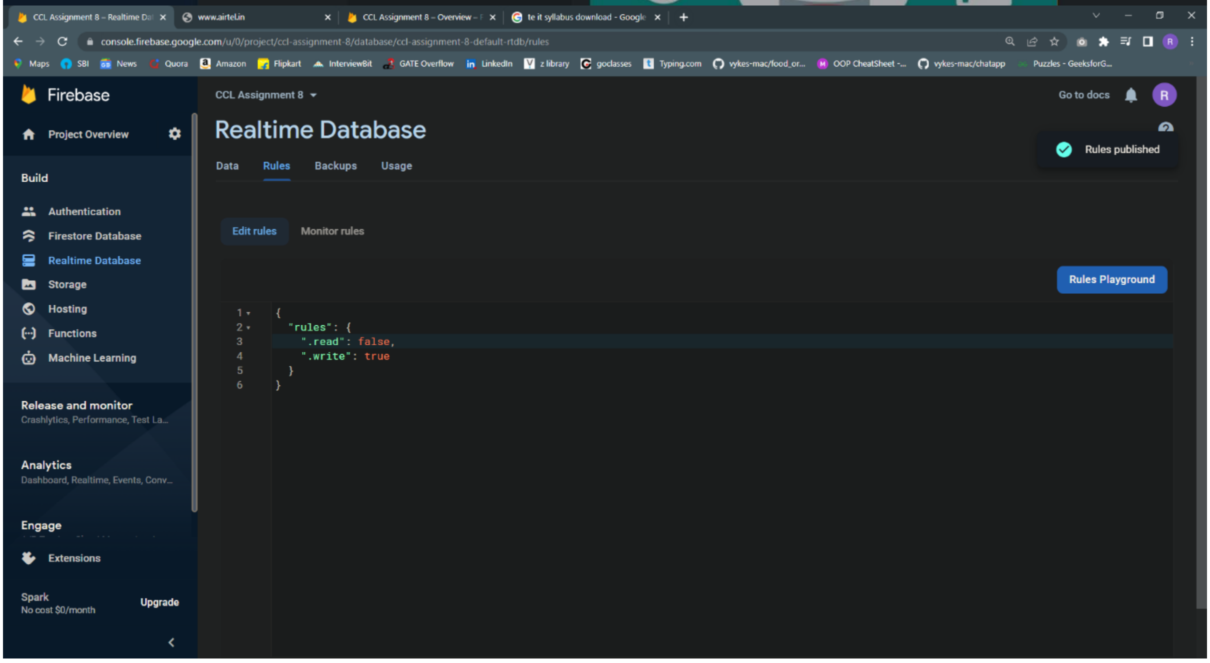
● Start in **locked mode**
Your data is private by default. Client read/write access will only be granted as specified by your security rules.

○ Start in **test mode**
Your data is open by default to enable quick setup. However, you must update your security rules within 30 days to enable long-term client read/write access.

```
{
  "rules": {
    ".read": false,
    ".write": false
  }
}
```

All third party reads and writes will be denied

Cancel　Enable

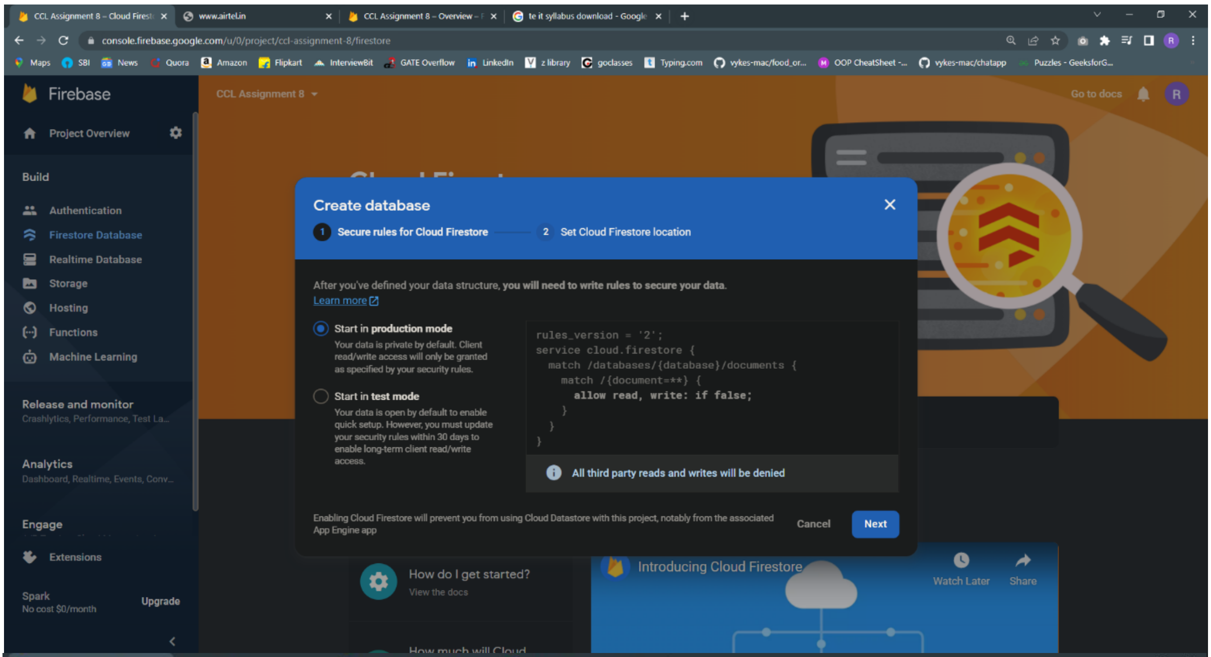


Realtime Database

Rules published

```
{
  "rules": {
    ".read": false,
    ".write": true
  }
}
```

3. Setting up Firestore Database:



Create database

1 Secure rules for Cloud Firestore — 2 Set Cloud Firestore location

After you've defined your data structure, **you will need to write rules to secure your data.**
Learn more

● Start in **production mode**
Your data is private by default. Client read/write access will only be granted as specified by your security rules.

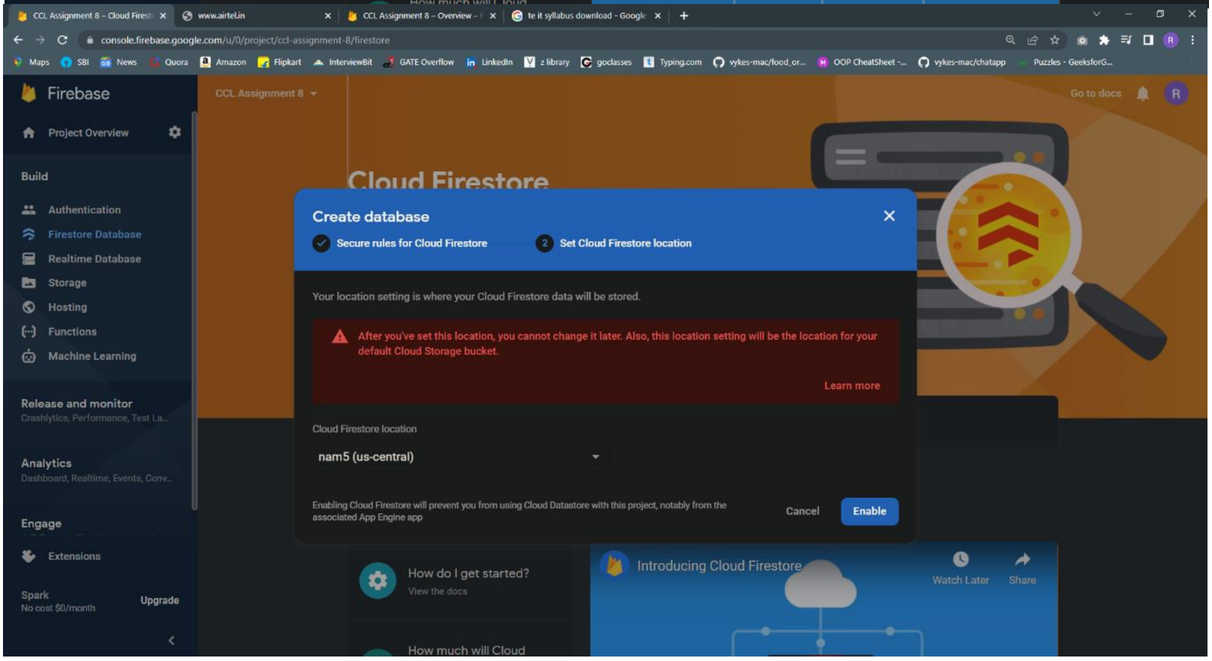
○ Start in **test mode**
Your data is open by default to enable quick setup. However, you must update your security rules within 30 days to enable long-term client read/write access.

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if false;
    }
  }
}
```

All third party reads and writes will be denied

Enabling Cloud Firestore will prevent you from using Cloud Datastore with this project, notably from the associated App Engine app

Cancel | Next



Create database

✓ Secure rules for Cloud Firestore — 2 Set Cloud Firestore location

Your location setting is where your Cloud Firestore data will be stored.

After you've set this location, you cannot change it later. Also, this location setting will be the location for your default Cloud Storage bucket.

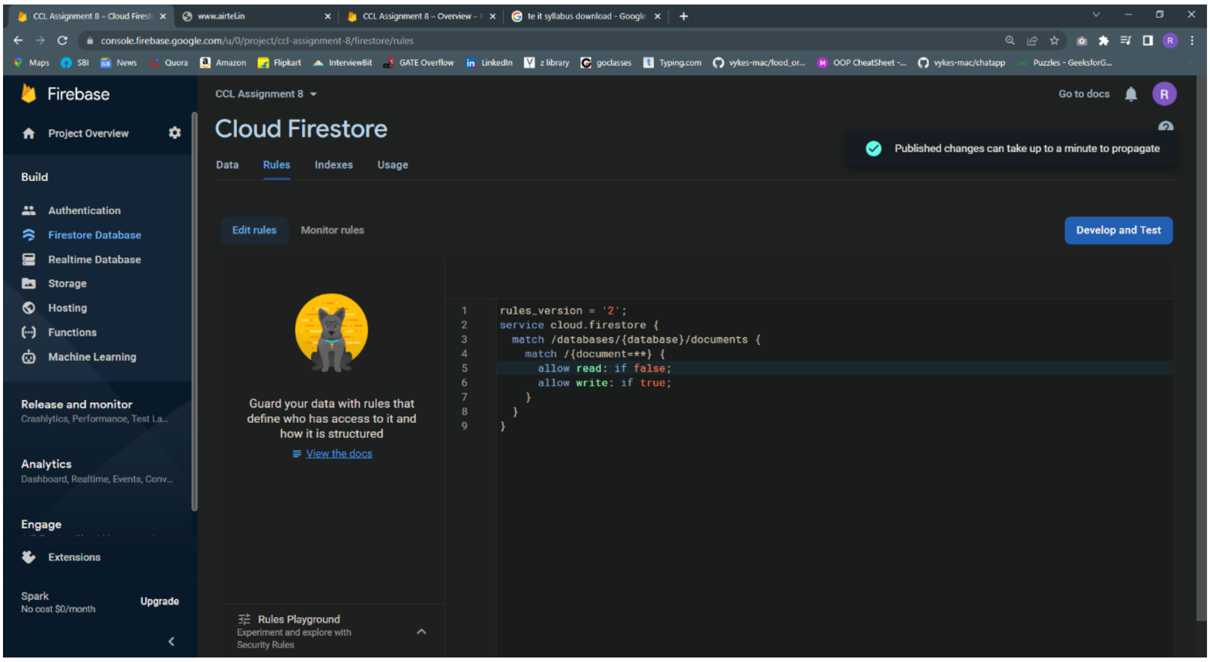
Learn more

Cloud Firestore location

nam5 (us-central)

Enabling Cloud Firestore will prevent you from using Cloud Datastore with this project, notably from the associated App Engine app

Cancel | Enable



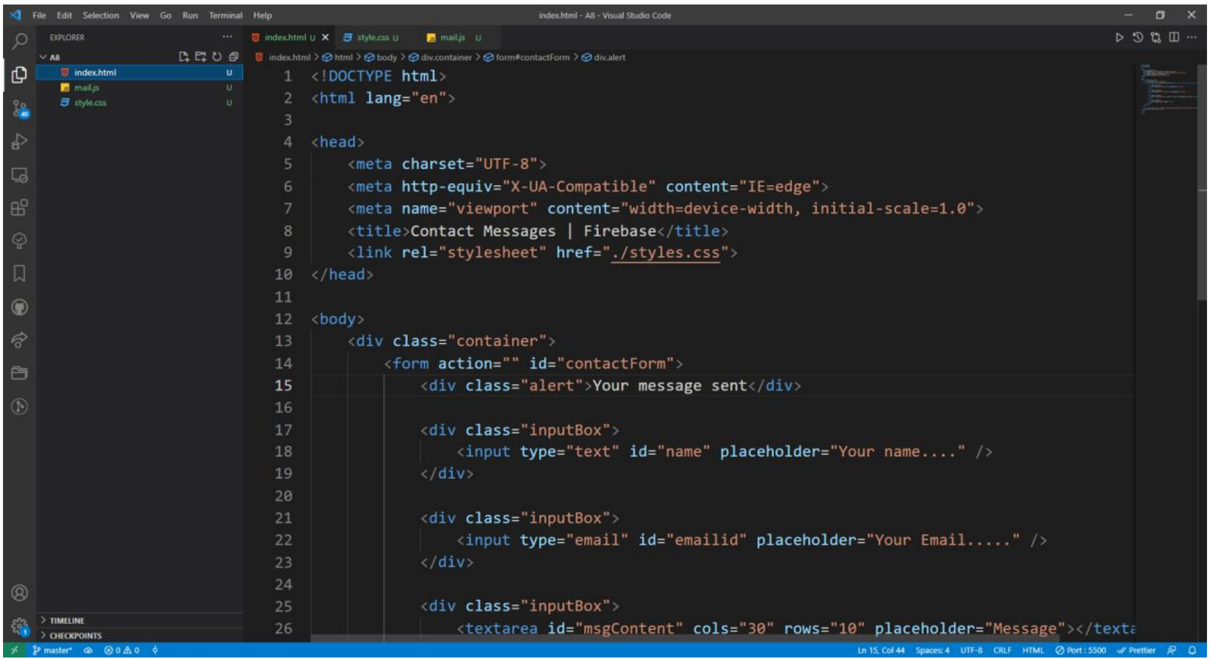
Published changes can take up to a minute to propagate

Guard your data with rules that define who has access to it and how it is structured

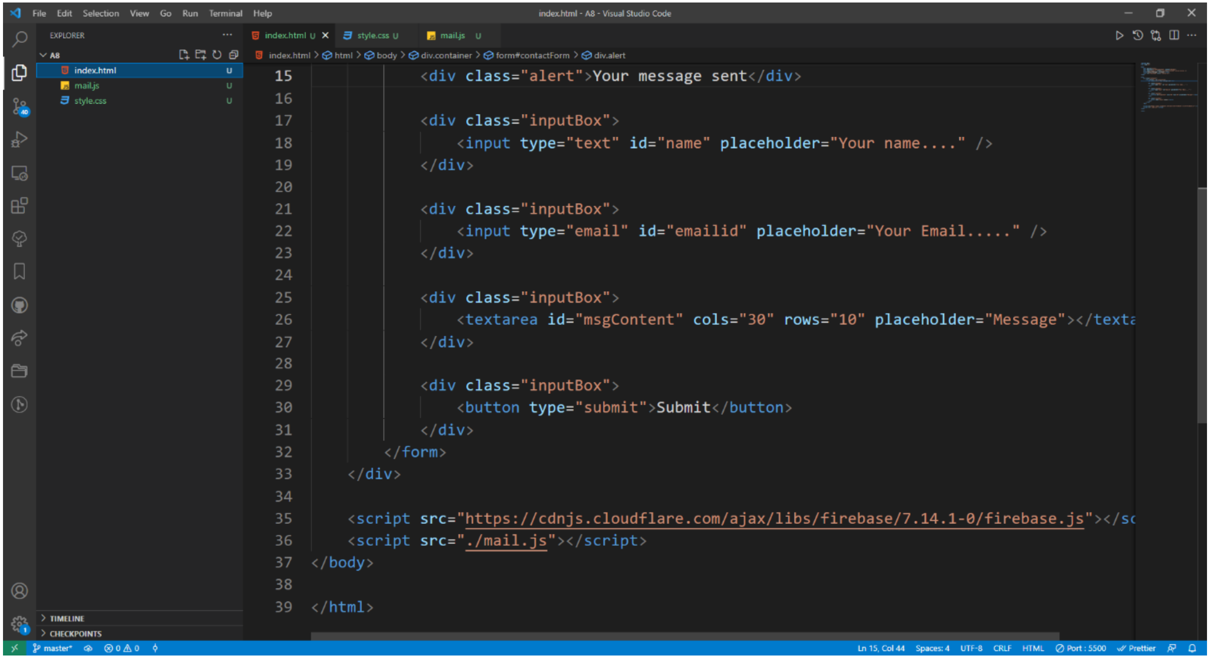
```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read: if false;
      allow write: if true;
    }
  }
}
```

Code Screenshots:

1. index.html:


File Edit Selection View Go Run Terminal Help
index.html - A8 - Visual Studio Code
EXPLORER
A8
index.html U
mail.js U
style.css U
index.html > html > body > div.container > form#contactForm > div.alert
TIMELINE
CHECKPOINTS
master*
Ln 15, Col 44 Spaces: 4 UTF-8 CRLF HTML Port : 5500 Prettier

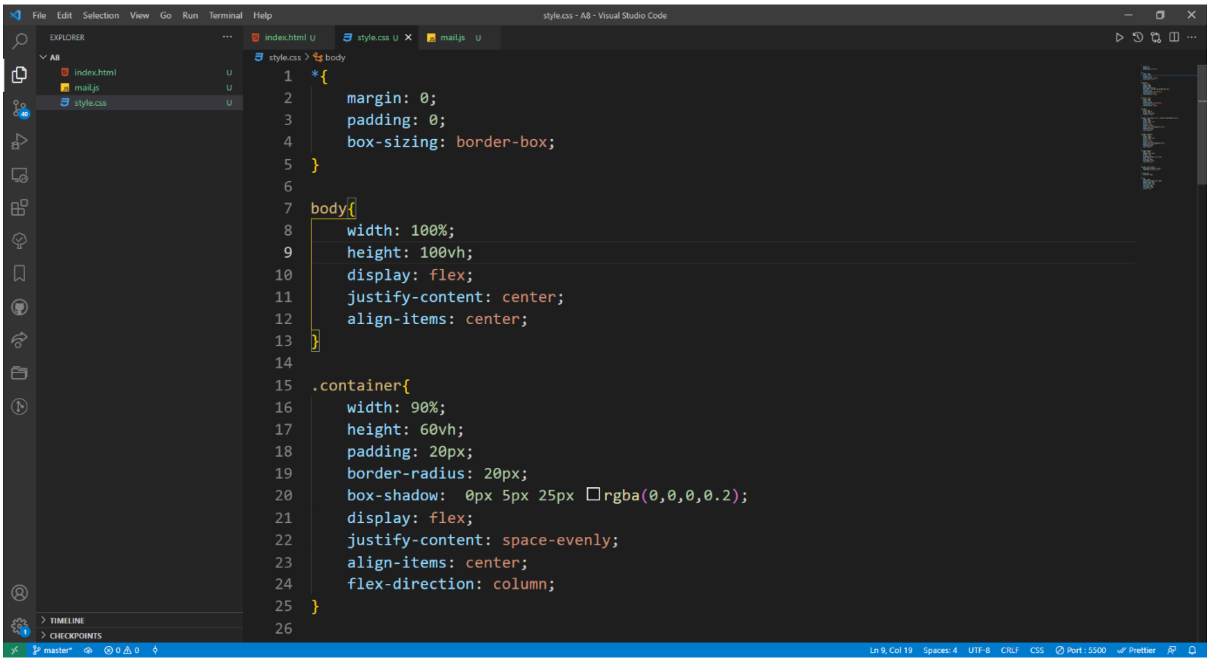
```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Contact Messages | Firebase</title>
    <link rel="stylesheet" href="./styles.css">
</head>

<body>
    <div class="container">
        <form action="" id="contactForm">
            <div class="alert">Your message sent</div>

            <div class="inputBox">
                <input type="text" id="name" placeholder="Your name...." />
            </div>

            <div class="inputBox">
                <input type="email" id="emailid" placeholder="Your Email....." />
            </div>

            <div class="inputBox">
                <textarea id="msgContent" cols="30" rows="10" placeholder="Message"></texta
```


File Edit Selection View Go Run Terminal Help
index.html - A8 - Visual Studio Code
EXPLORER
A8
index.html U
mail.js U
style.css U
index.html > html > body > div.container > form#contactForm > div.alert
TIMELINE
CHECKPOINTS
master*
Ln 15, Col 44 Spaces: 4 UTF-8 CRLF HTML Port : 5500 Prettier

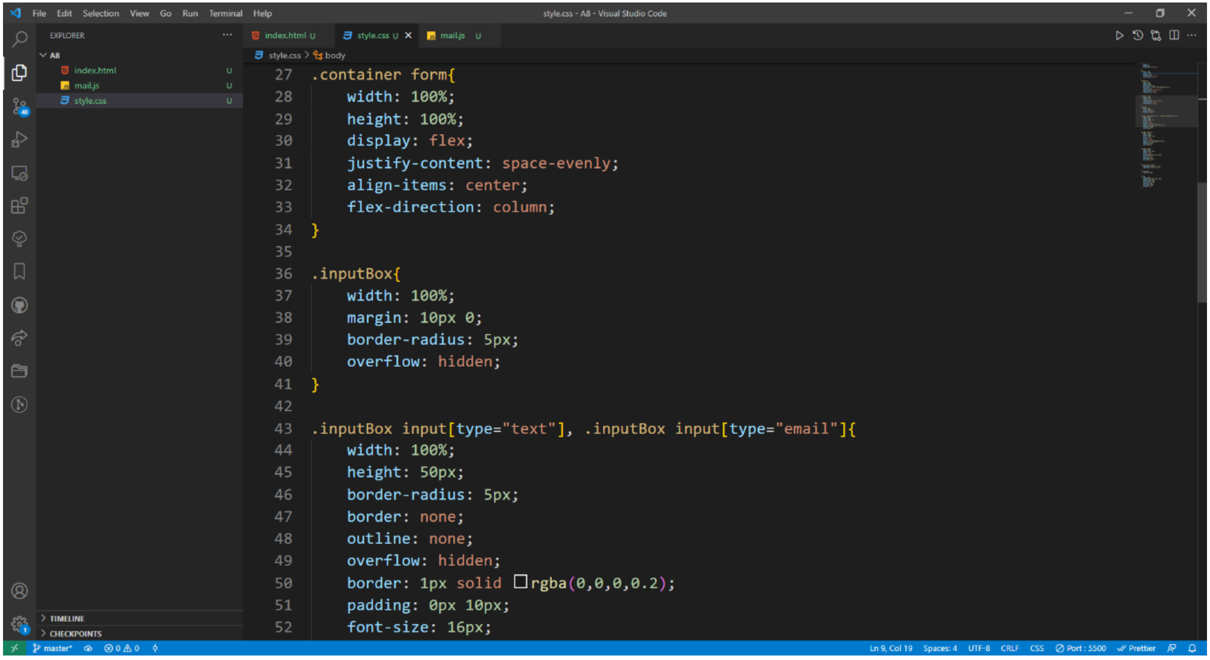
```html
            <div class="alert">Your message sent</div>

            <div class="inputBox">
                <input type="text" id="name" placeholder="Your name...." />
            </div>

            <div class="inputBox">
                <input type="email" id="emailid" placeholder="Your Email....." />
            </div>

            <div class="inputBox">
                <textarea id="msgContent" cols="30" rows="10" placeholder="Message"></texta
```

```html
            </div>

            <div class="inputBox">
                <button type="submit">Submit</button>
            </div>
        </form>
    </div>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/firebase/7.14.1-0/firebase.js"></sc
    <script src="./mail.js"></script>
</body>

</html>
```

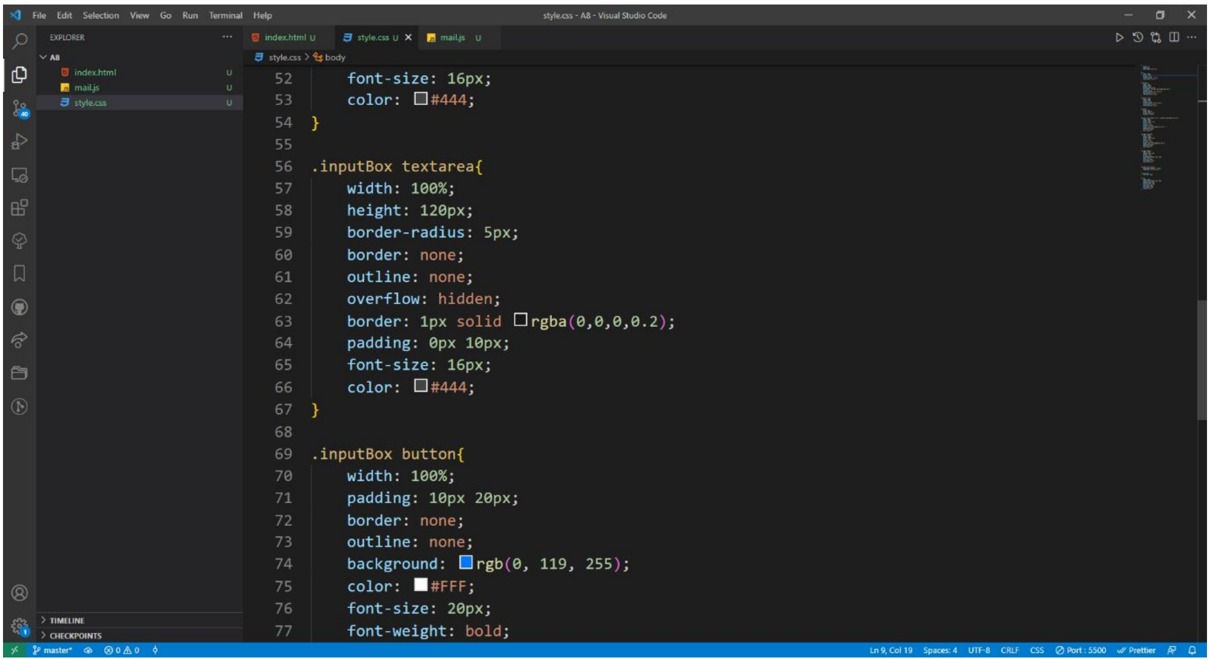
style.css:


File Edit Selection View Go Run Terminal Help
style.css - A8 - Visual Studio Code
EXPLORER
A8
index.html U
mail.js U
style.css U
index.html U
style.css U
mail.js U
style.css > body
TIMELINE
CHECKPOINTS
master*
Ln 9, Col 19 Spaces: 4 UTF-8 CRLF CSS Port : 5500 Prettier

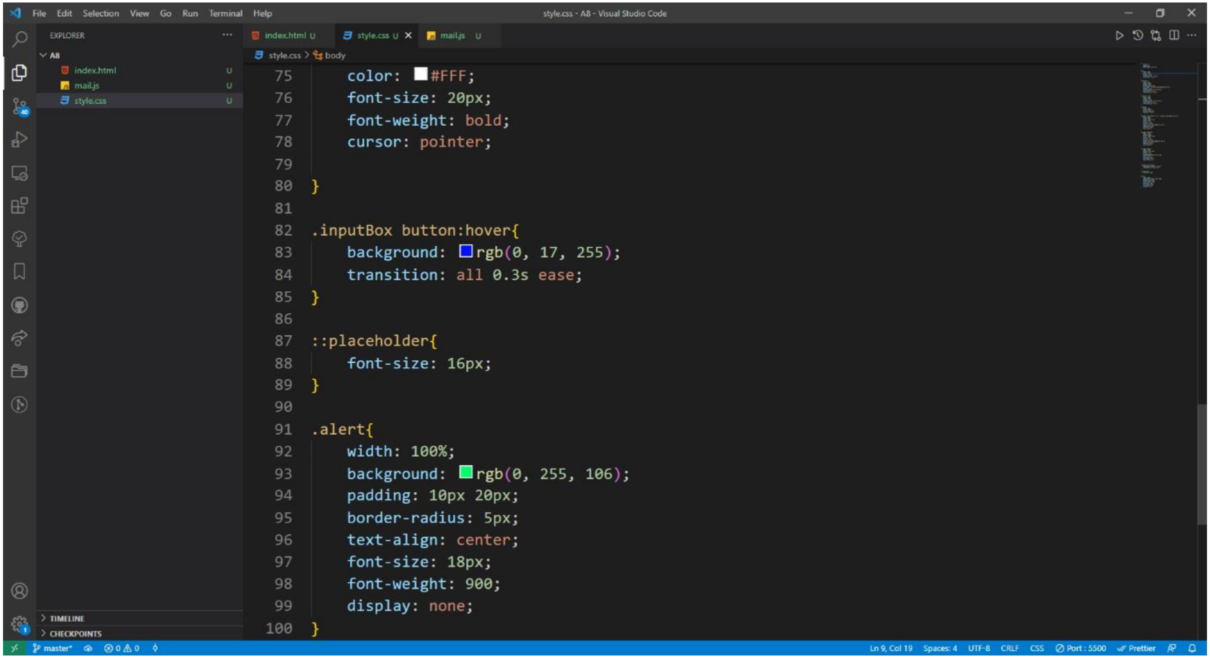
```css
*{
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body{
    width: 100%;
    height: 100vh;
    display: flex;
    justify-content: center;
    align-items: center;
}

.container{
    width: 90%;
    height: 60vh;
    padding: 20px;
    border-radius: 20px;
    box-shadow:  0px 5px 25px □rgba(0,0,0,0.2);
    display: flex;
    justify-content: space-evenly;
    align-items: center;
    flex-direction: column;
}

```


File Edit Selection View Go Run Terminal Help
style.css - A8 - Visual Studio Code
EXPLORER
A8
index.html U
mail.js U
style.css U
index.html U
style.css U
mail.js U
style.css > body
TIMELINE
CHECKPOINTS
master*
Ln 9, Col 19 Spaces: 4 UTF-8 CRLF CSS Port : 5500 Prettier

```css
.container form{
    width: 100%;
    height: 100%;
    display: flex;
    justify-content: space-evenly;
    align-items: center;
    flex-direction: column;
}

.inputBox{
    width: 100%;
    margin: 10px 0;
    border-radius: 5px;
    overflow: hidden;
}

.inputBox input[type="text"], .inputBox input[type="email"]{
    width: 100%;
    height: 50px;
    border-radius: 5px;
    border: none;
    outline: none;
    overflow: hidden;
    border: 1px solid □rgba(0,0,0,0.2);
    padding: 0px 10px;
    font-size: 16px;
```

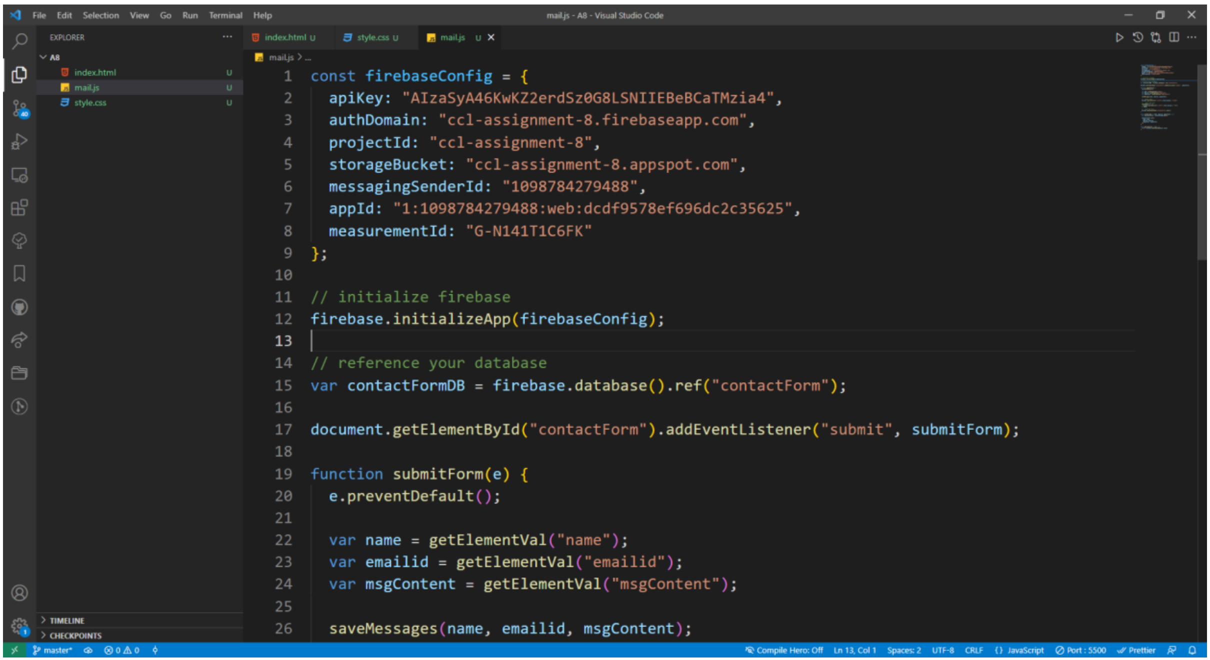

File Edit Selection View Go Run Terminal Help
style.css - A8 - Visual Studio Code
EXPLORER
A8
index.html U
mail.js U
style.css U
style.css > body
TIMELINE
CHECKPOINTS
master*
Ln 9, Col 19 Spaces: 4 UTF-8 CRLF CSS Port : 5500 Prettier

```css
    font-size: 16px;
    color: #444;
}

.inputBox textarea{
    width: 100%;
    height: 120px;
    border-radius: 5px;
    border: none;
    outline: none;
    overflow: hidden;
    border: 1px solid rgba(0,0,0,0.2);
    padding: 0px 10px;
    font-size: 16px;
    color: #444;
}

.inputBox button{
    width: 100%;
    padding: 10px 20px;
    border: none;
    outline: none;
    background: rgb(0, 119, 255);
    color: #FFF;
    font-size: 20px;
    font-weight: bold;
```

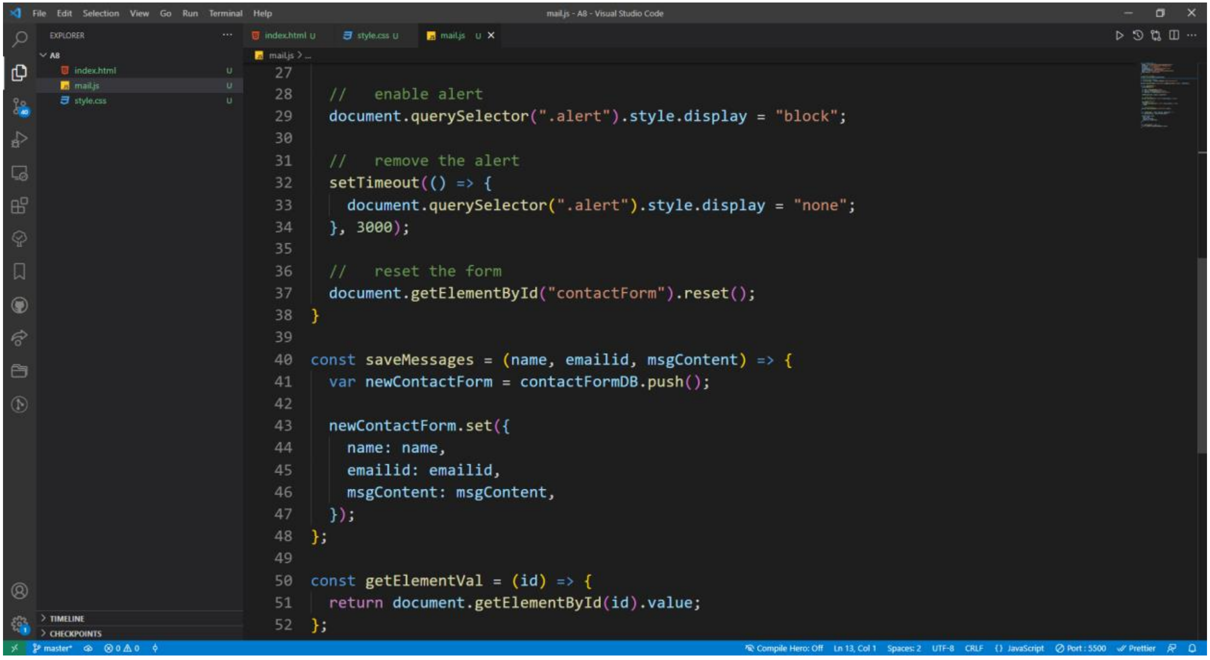

File Edit Selection View Go Run Terminal Help
style.css - A8 - Visual Studio Code
EXPLORER
A8
index.html U
mail.js U
style.css U
style.css > body
TIMELINE
CHECKPOINTS
master*
Ln 9, Col 19 Spaces: 4 UTF-8 CRLF CSS Port : 5500 Prettier

```css
    color: #FFF;
    font-size: 20px;
    font-weight: bold;
    cursor: pointer;

}

.inputBox button:hover{
    background: rgb(0, 17, 255);
    transition: all 0.3s ease;
}

::placeholder{
    font-size: 16px;
}

.alert{
    width: 100%;
    background: rgb(0, 255, 106);
    padding: 10px 20px;
    border-radius: 5px;
    text-align: center;
    font-size: 18px;
    font-weight: 900;
    display: none;
}
```

mail.js:



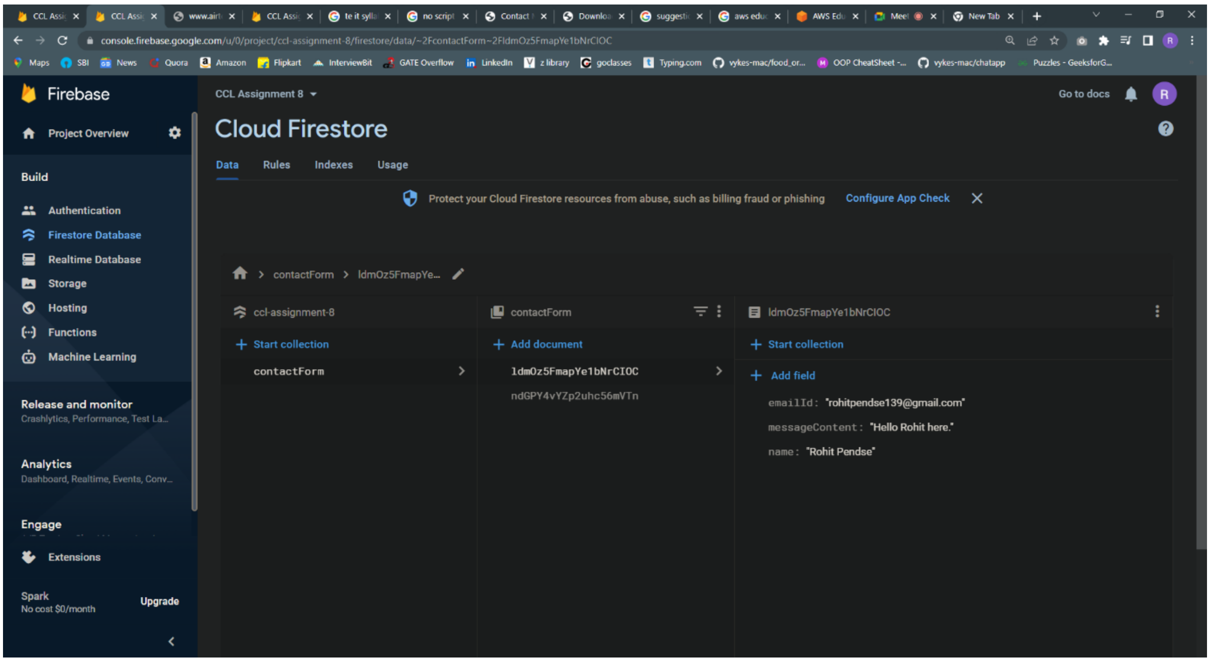
```javascript
const firebaseConfig = {
  apiKey: "AIzaSyA46KwKZ2erdSz0G8LSNIIEBeBCaTMzia4",
  authDomain: "ccl-assignment-8.firebaseapp.com",
  projectId: "ccl-assignment-8",
  storageBucket: "ccl-assignment-8.appspot.com",
  messagingSenderId: "1098784279488",
  appId: "1:1098784279488:web:dcdf9578ef696dc2c35625",
  measurementId: "G-N141T1C6FK"
};

// initialize firebase
firebase.initializeApp(firebaseConfig);

// reference your database
var contactFormDB = firebase.database().ref("contactForm");

document.getElementById("contactForm").addEventListener("submit", submitForm);

function submitForm(e) {
  e.preventDefault();

  var name = getElementVal("name");
  var emailid = getElementVal("emailid");
  var msgContent = getElementVal("msgContent");

  saveMessages(name, emailid, msgContent);
```



```javascript

  //   enable alert
  document.querySelector(".alert").style.display = "block";

  //   remove the alert
  setTimeout(() => {
    document.querySelector(".alert").style.display = "none";
  }, 3000);

  //   reset the form
  document.getElementById("contactForm").reset();
}

const saveMessages = (name, emailid, msgContent) => {
  var newContactForm = contactFormDB.push();

  newContactForm.set({
    name: name,
    emailid: emailid,
    msgContent: msgContent,
  });
};

const getElementVal = (id) => {
  return document.getElementById(id).value;
};
```

Output Screenshots:

Firebase Outputs:



console.firebase.google.com/u/0/project/ccl-assignment-8/firestore/data/~2FcontactForm~2FldmOz5FmapYe1bNrCIOC

Protect your Cloud Firestore resources from abuse, such as billing fraud or phishing

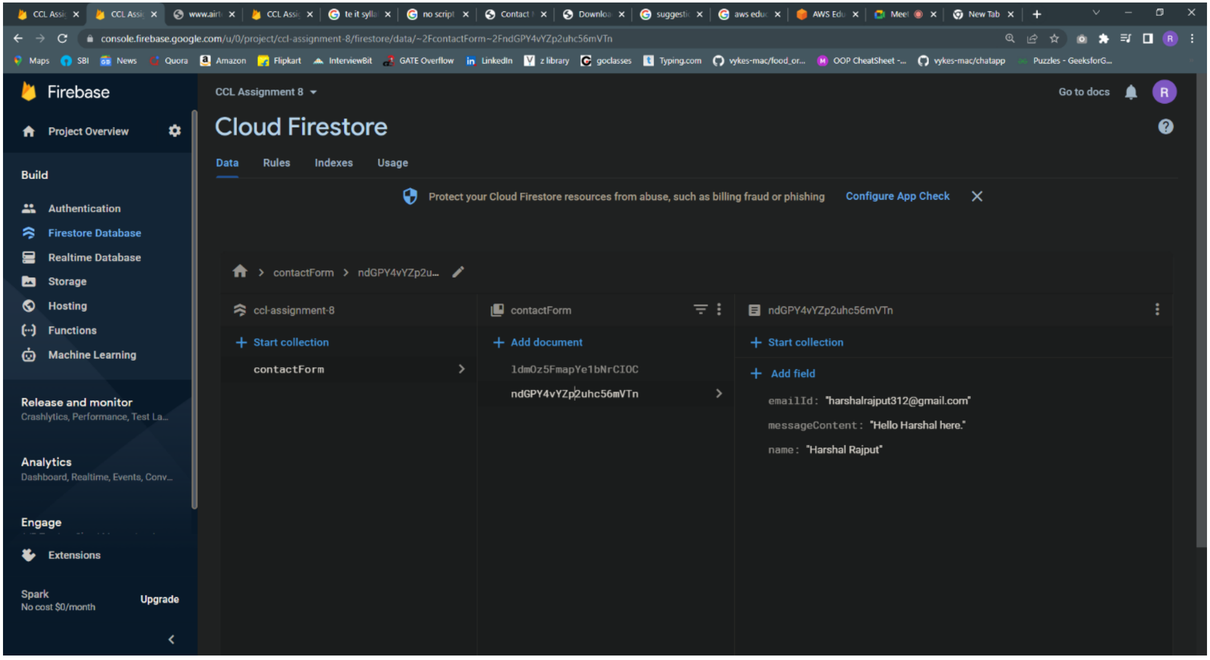
contactForm > ldmOz5FmapYe...

ccl-assignment-8: contactForm

contactForm: 1dmOz5FmapYe1bNrCIOC, ndGPY4vYZp2uhc56mVTn

ldmOz5FmapYe1bNrCIOC

```
emailId: "rohitpendse139@gmail.com"
messageContent: "Hello Rohit here."
name: "Rohit Pendse"
```



console.firebase.google.com/u/0/project/ccl-assignment-8/firestore/data/~2FcontactForm~2FndGPY4vYZp2uhc56mVTn

Protect your Cloud Firestore resources from abuse, such as billing fraud or phishing

contactForm > ndGPY4vYZp2u...

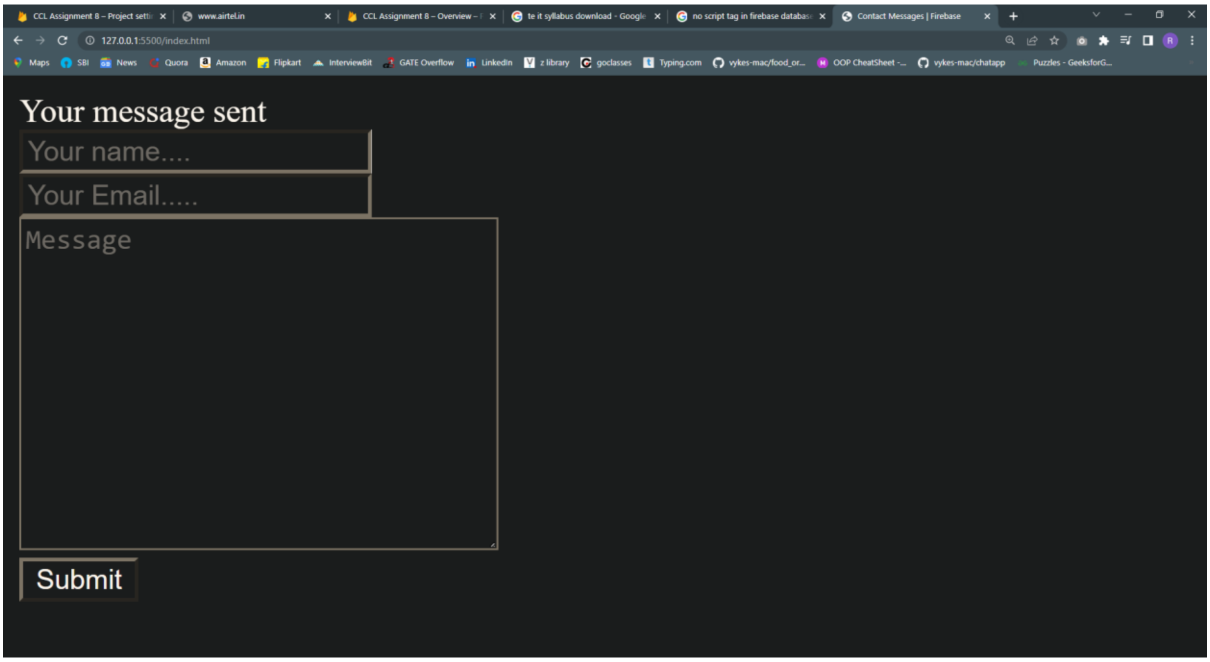
ccl-assignment-8: contactForm

contactForm: 1dmOz5FmapYe1bNrCIOC, ndGPY4vYZp2uhc56mVTn

ndGPY4vYZp2uhc56mVTn

```
emailId: "harshalrajput312@gmail.com"
messageContent: "Hello Harshal here."
name: "Harshal Rajput"
```

UI screenshots:



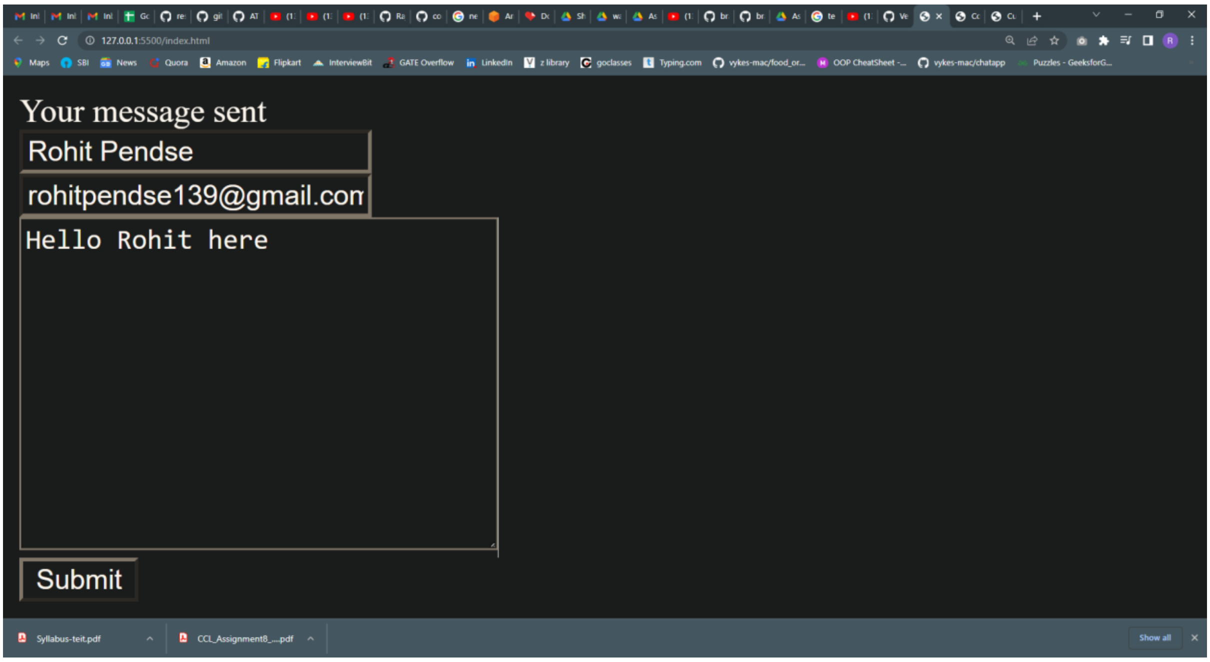
Your message sent

Your name....

Your Email.....

Message

Submit



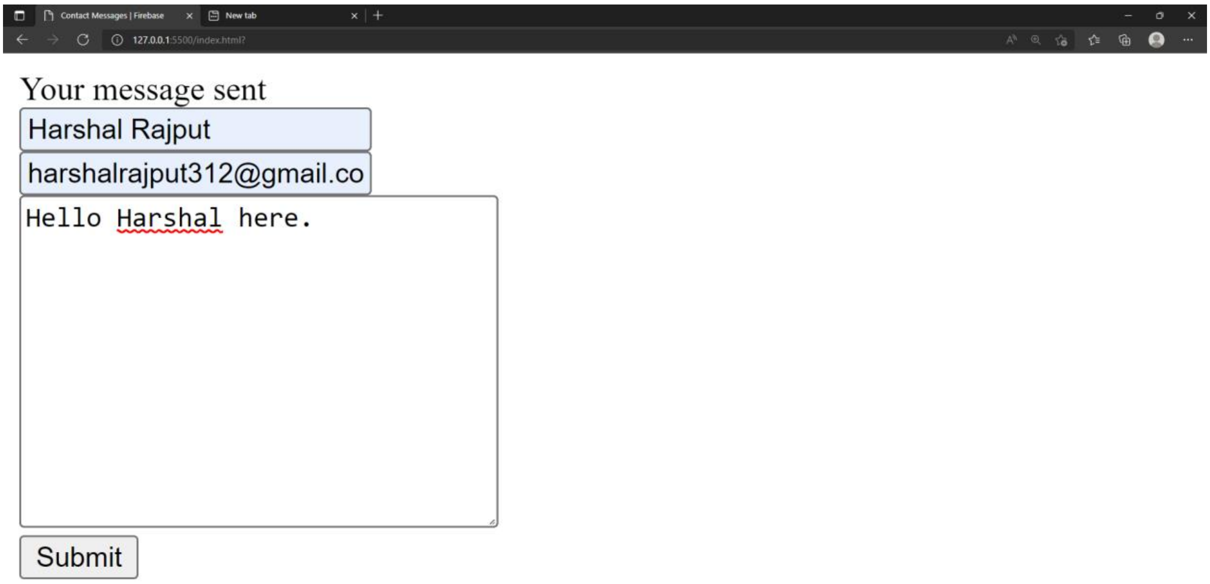
Your message sent

Rohit Pendse

rohitpendse139@gmail.com

Hello Rohit here

Submit



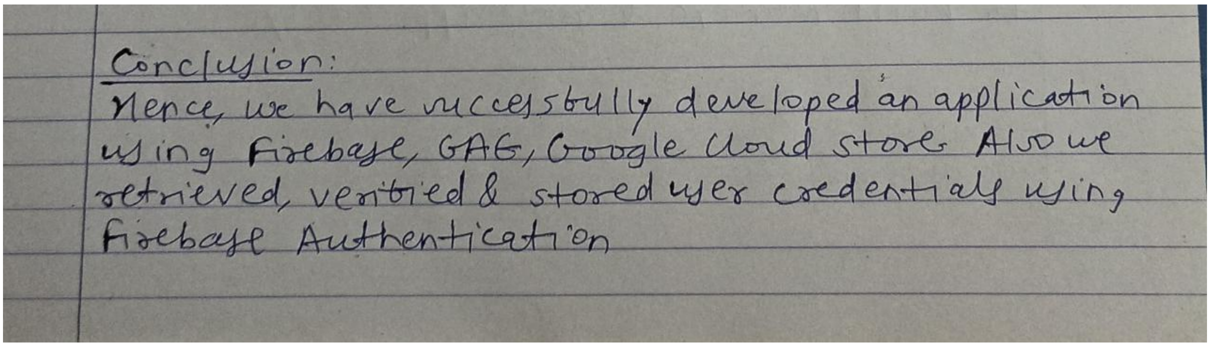
Your message sent

Harshal Rajput

harshalrajput312@gmail.co

Hello Harshal here.

Conclusion:



Conclusion:

Hence, we have succesfully developed an application using Firebase, GAE, Google Cloud store. Also we retrieved, verified & stored user credentials using Firebase Authentication