

KEC-CIRCULAR APP



Name: JANARTHANAN T

Roll No: 20ITR038

CLASS: IT-A

Course Name: Software Engineering

Course Code: 20ITT53

Title: KEC-CIRCULAR APP

Aim:

The main aim of the project is to prevent fake circulars being spreaded inside the college also a common app for all students and faculties in KEC to view circulars and send notification to them to remind them.

TOOLS USED:

Website:

- Front-End: EJS
- Back-End: Node JS
- Database: Mongo Db

App:

- Front-End: React- Native
- Back-End: Node JS
- Database: Mongo Db



CODING:

```
server.js

const express = require('express')

const bodyParser = require('body-parser');

const mongoose = require('mongoose');

const ejsMate = require('ejs-mate')

const methodOverride = require('method-override')

const path = require("path")

const cookieParser = require("cookie-parser");

const sessions = require('express-session');

const dotenv = require('dotenv')

const flash = require('connect-flash');

const User = require("./models/user");

const Circular = require("./models/circular")

const Constant = require("./models/constant")

const { isLoggedIn } = require("./middleware/auth")

const userRoutes = require('./routes/user')

const circularRoutes = require('./routes/circular')

const notificationRoutes=require('./routes/notification')const app = express()

app.use(express.static(path.join(__dirname, "/public")))

app.use(methodOverride('_method'))

app.engine('ejs', ejsMate)

dotenv.config()

app.use(express.json({ extended: true }))

app.use(express.urlencoded({ extended: true }));

app.use(flash());

const oneDay = 1000 * 60 * 60 * 24;

app.use(sessions({

  secret: "webpirates",
```



```

    saveUninitialized: true,
    cookie: { maxAge: oneDay },
    resave: false
  ));
  app.use(async (req, res, next) => {
    res.locals.success=req.flash('success');
    res.locals.error=req.flash('error');
    res.locals.currentUser = req.session._id;
    next()
  })
  app.use('/user', userRoutes)
  app.use('/circular', circularRoutes)
  app.use('/api',notificationRoutes)
  app.set('view engine', 'ejs');
  app.use(bodyParser.json());
  app.use(bodyParser.urlencoded({ extended: true }));
  app.use(cookieParser());
  mongoose.connect(process.env.DB).then(() => {
    console.log('Db connection open')
  }).catch(err => {
    console.log(err.message, 'oops err');
  });
  var session; //to make variable available in all place
  app.get("/", (req, res) => {
    session = req.session;
    if (session._id) {
      res.redirect('/circular/all/web')
    }else {
      req.flash("success","Hii")
      res.render('auth_page/login')
    }
  })

```



```
  })  
  const PORT = process.env.PORT || 3000  
  app.listen(PORT, () => {  
    console.log(`Server is running at Port ${PORT}`)  
  })
```

User.js

```
const express=require('express')  
  
const { signUp, login,webSignUp,webLogin,renderLogin,renderRegister, getsignup,logout,  
deleteDeviceId } = require('../controllers/user')  
  
const { change_password,change_password_request,link_from_mail,forgotten_password }  
=require('../controllers/forgotten_password.controller')  
  
const { otp_sendEmail,resetpassword_sendEmail } = require('../controllers/mail')  
  
const router = express.Router();  
  
router.get('/login',renderLogin)  
  
router.get('/register',renderRegister)  
  
router.get('/getsignup',getsignup)  
  
router.post('/signup',signUp)  
  
router.post('/login',login)  
  
router.post('/signup/web',webSignUp)  
  
router.post('/login/web',webLogin)  
  
router.post('/otp',otp_sendEmail)  
  
router.post('/delete',deleteDeviceId)  
  
router.get('/logout',logout)  
  
router.get("/forgotten-password",forgotten_password)  
  
router.get('/forgotten-password/:userId/:token',link_from_mail)  
  
router.post('/forgotten-password',change_password_request)  
  
router.post('/forgotten-password/:userId/:token',change_password)  
  
module.exports=router;  
  
notification.js  
  
const express=require('express')
```



```
const router = express.Router();

const pushNotificationController=require("../controllers/push_notification.controllers")

router.get('/sendnotification',pushNotificationController.SendNotification)

router.post('/sendnotificationToDevice',pushNotificationController.SendNotificationToDevice)

module.exports=router;
```

circular.js

```
const express=require('express')

const {postCircular,renderCircular,getAllCircular,deleteCircular} =
require('../controllers/circular.js')

const router = express.Router();

const { storage, fileFilter } = require("../multter/upload")

const multer = require('multer');

const upload = multer({ limits: { fileSize: 2097152 },fileFilter: fileFilter, storage: storage })

const {isLoggedIn}=require("../middleware/auth")

router.get('/',isLoggedIn,renderCircular)

router.post('/',isLoggedIn,upload.single('pdf'),postCircular)

router.get('/all/:platform',getAllCircular)

router.get('/:id',deleteCircular)

module.exports=router;
```

CONTROLLERS:

Circular.js

```
const Circular = require('../models/circular.js');

const user = require('../models/user.js');

const notify = require('../controllers/push_notification.controllers')

const Constant = require('../models/constant.js')

module.exports.postCircular = async(req,res) =>{
```



```

try{
  const result = {
    postedOn: Date.now(),
    title: req.body.title,
    batch: req.body.batch,
    dept: req.body.dept,
    number: req.body.number,
    filePath: req.file.path.substring(6),
    postedBy: req.session._id
  }
  var userlist;
  if (req.body.dept == 'all' && req.body.batch == 'all') {
    userlist = await user.find({});
    console.log(userlist+"all")
  }
  else if (req.body.dept == 'all' || req.body.batch == 'all') {
    userlist = await user.find({ $or: [{ department: { $in: req.body.dept } }, { batch: { $in: req.body.batch } } ] });
    console.log(userlist+"any one")
  }
  else {
    userlist = await user.find({ $and: [{ department: { $in: req.body.dept } }, { batch: { $in: req.body.batch } } ] });
    console.log(userlist+"none")
  }
  //need to work for push notification
  devices = []
  userlist.forEach((ele) => {
    if (ele.deviceId != '-')
    devices.push(ele.deviceId)
  })

```



```

    notify.pushnotify(devices);

    const circular = await new Circular(result);

    await circular.save()

    req.flash('success','Circular Posted successfully')

    res.redirect('/');

  } catch (err) {

    console.log(err.message)

  }}

module.exports.renderCircular = async (req, res) => {

  var increment;

  var year=new Date().getFullYear()

  const batchYear=[]

  const department=await Constant.findOne({});

  for(increment=-4;increment<=4;increment++){

    batchYear.push(year-increment);

  }

  res.render("circular_page/add_circular.ejs",{batchYear,department})

}

module.exports.deleteCircular= async(req,res)=>{

  const {id}=req.params

  try {

    const circular=await Circular.findByIdAndDelete(id)

    req.flash('success','Circular has been deleted successfully')

    res.redirect('/circular/all/web')

  } catch (error) {

    console.log(error)

  }}

module.exports.getAllCircular = async (req, res) => {

  const currentYear = new Date().getFullYear();

```




```

const currentMonth = new Date().getMonth() + 1;
const currentDate = new Date().getDate();

//today timestamp

const timestamp = [currentYear, currentMonth <= 9 ? '0' + currentMonth : currentMonth,
currentDate <= 9 ? '0' + currentDate : currentDate].join('-');

const today = new Date(timestamp)

//yesterday timestamp

const previous = new Date(today.getTime());
previous.setDate(previous.getDate() - 1);
const yesterday = previous;

try {
  //query

  var yesterdayCircular = await Circular.find({ postedOn: { $gte: yesterday, $lt: today }
})

  yesterdayCircular=yesterdayCircular.sort((a,b)=>b.number-a.number);

  var todayCircular = await Circular.find({ postedOn: { $gt: today } })

  todayCircular = todayCircular.sort((a,b)=>b.number-a.number);

  var allCircular = await Circular.find({ postedOn: { $lt: yesterday } })

  allCircular = allCircular.sort((a,b)=>b.number-a.number)

  //seperating according to months for all circular

  monthwise = [{ "title": "January", "data": [] },
  { "title": "February", "data": [] },
  { "title": "March", "data": [] },
  { "title": "April", "data": [] },
  { "title": "May", "data": [] },
  { "title": "June", "data": [] },
  { "title": "July", "data": [] },
  { "title": "August", "data": [] },
  { "title": "September", "data": [] },
  { "title": "October", "data": [] },
  { "title": "November", "data": [] },

```



```

    { "title": "December", "data": [] } ]
    allCircular.map((ele) => {
        index = ele.postedOn.getMonth()
        monthwise[index].data.push(ele)
    })
    req.params.platform == 'web' ?
        res.render('circular_page/view_allcircular', { allCircular, yesterdayCircular,
todayCircular, monthwise }) :
        res.status(200).json({ allCircular, yesterdayCircular, todayCircular, monthwise })
    } catch (err) {
        console.log(err)
        res.status(500).json('Something went wrong...')
    }
}

```

User.js

```

const mongoose = require('mongoose')
const jwt = require('jsonwebtoken')
const bcrypt = require('bcryptjs')
const User = require('../models/user')
const Constant=require("../models/constant")
var session;

module.exports.getsignup=async (req,res)=>{
    const department=await Constant.findOne({});
    res.status(200).json({ department })
}

module.exports.signUp = async (req, res) => {
    const { name, rollno, email, department, password, batch, type,deviceId} = req.body
    try {
        const existinguser = await User.findOne({ email })

```



```

    if (existinguser) {
        return res.status(400).json({ message: 'User already found..' })
    }

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

    const newUser = new User({ name, rollno, email, department, batch, type, deviceId,
password: hashPassword })

    await newUser.save();

    const token = jwt.sign({ email: newUser.email, id: newUser._id }, 'token', { expiresIn:
'1h' })

    res.status(200).json({ result: newUser, token })

} catch (err) {
    console.log(err.message)
    res.status(500).json('Something went wrong...')
}
}

```

```

module.exports.webSignUp = async(req,res) =>{
    const { name,department,password,email,type } = req.body
    try{
        let femail_pattern=/^([a-z]+\.)\.[a-z]{2,5}\.([a-z]+\.)\.[a-z]{2,5}$/;
        let result1=femail_pattern.test(email)
        if(!result1 &&!email.endsWith('kongu.edu')){
            req.flash('error','Invalid email')
            return res.redirect('/user/register')
        }
        var isAdmin=false,isDeptAdmin=false
        var rollno=""
        const existinguser = await User.findOne({ email })
        if (existinguser) {
            req.flash('error','User found already')
            return res.redirect('/user/register') }
    }
}

```



```

    if(type==='admin'){
        isAdmin=true
        rollno='admin'
    }else if (type==='department-admin'){
        isDeptAdmin=true
        rollno=`${department}admin`
    }

    const hashPassword = await bcrypt.hash(password, 12);
    const newUser = new User({ name, email, rollno,type,
    department,isAdmin,isDeptAdmin, password: hashPassword })
    await newUser.save();
    req.session._id = newUser._id;
    res.locals.currentUser = newUser._id
    req.flash('success','Account created')
    res.redirect('/')
} catch(err){
}
}

module.exports.webLogin = async(req,res) =>{
    const { email, password } = req.body;
    const existinguser = await User.findOne({ email })
    if (!existinguser) {
        req.flash('error','User not found')
        return res.redirect('/user/login')
    }
    const isPasswordCrt = await bcrypt.compare(password, existinguser.password)
    if (!isPasswordCrt) {
        req.flash('error','Invalid Credentials')
        return res.redirect('/user/login')
    }
    if(!existinguser.isAdmin){

```



```

    req.flash('error','Unauthorised access')

    return res.redirect('/user/login')
  }
  session = req.session;
  req.session._id = existinguser._id
  req.flash('success','Login Successfull')
  res.redirect('/')
}

module.exports.login = async (req, res) => {
  const { email, password, deviceId } = req.body;
  try {
    const existinguser = await User.findOne({ email })
    if (!existinguser) {
      console.log("User not found...");
      return res.status(404).json({ message: "User not found..." })
    }
    const isPasswordCrt = await bcrypt.compare(password, existinguser.password)
    if (!isPasswordCrt) {
      return res.status(400).json({ message: "Invalid credentials" })
    }
    existinguser.deviceId=deviceId
    await existinguser.save()

    const token = jwt.sign({ email: existinguser.email, id: existinguser._id }, 'token', {
      expiresIn: '48h' })
    res.status(200).json({ result: existinguser, token })
  } catch(err) {
    console.log(err.message)
    res.status(500).json(err.message)
  }
}

module.exports.renderLogin = async(req,res) =>{

```



```

    res.render('auth_page/login.ejs')
  }
  module.exports.renderRegister = async(req,res) =>{
    const department=await Constant.findOne({ });
    res.render('auth_page/signup.ejs',{ department})
  }
  module.exports.logout=async(req,res)=>{
    if (req.session) {
      req.session.destroy();
    }
    res.redirect("/user/login")
  }
  module.exports.deleteDeviceId = async(req,res)=>{
    const {id}=req.body
    try {
      const user=await User.findById(id)
      user.deviceId = '-'
      await user.save()
      res.status(200).send('Logout successfull')
    } catch (error) {
      res.status(500).send(error)
    }
  }
}

```

CLIENT:

Index.js

```

import {AppRegistry} from 'react-native';
import App from './App';
import {name as appName} from './app.json';
AppRegistry.registerComponent(appName, () => App );

```



App.js

```
import React, { useEffect } from 'react';
import { SafeAreaView, StyleSheet } from 'react-native';
import AppNavigation from './src/screens/AppNavigation';
import { Provider } from 'react-redux';
import Reducers from './src/reducers';
import { createStore, applyMiddleware, compose } from 'redux';
import thunk from 'redux-thunk';

const store = createStore(Reducers, compose(applyMiddleware(thunk)))

import OneSignal from 'react-native-onesignal';
OneSignal.setAppId('1e8e3e87-0155-46a5-aa28-545f0512a0fb');

const App = () => {
  return (
    <Provider store={store}>
      <SafeAreaView style={styles.root}>
        <AppNavigation />
      </SafeAreaView>
    </Provider> );
};

const styles = StyleSheet.create({
  root: {
    flex: 1,
    backgroundColor: "#F9F7F7"
  }
});

export default App;
```

AppNavigation.js

```
import React from 'react'
import { NavigationContainer } from '@react-navigation/native';
import { createNativeStackNavigator } from '@react-navigation/native-stack';
```



```

import Home from '../Home';
import Circular from '../Circular';
import Auth from '../Auth';
import VerifyOTP from '../components/VerifyOTP';
import ForgetPassword from '../components/ForgetPassword/ForgetPassword';
import CircularPreview from '../CircularPreview';
const Stack = createNativeStackNavigator();
const AppNavigation = () => {
  return (
    <NavigationContainer>
      <Stack.Navigator screenOptions={{ headerShown: false }} initialRouteName="Home">
        <Stack.Screen name='Home' component={ Home } />
        <Stack.Screen name='Circular' component={ Circular } />
        <Stack.Screen name='CircularPreview' component={ CircularPreview } />
        <Stack.Screen name='Auth' component={ Auth } />
        <Stack.Screen name="VerifyOtp" component={ VerifyOTP } />
        <Stack.Screen name='Forget' component={ ForgetPassword } />
      </Stack.Navigator>
    </NavigationContainer>
  );
}
export default AppNavigation

```

Auth.js

```

import React, { useEffect, useState } from 'react'
import { Text, View, Image, ScrollView, StyleSheet } from 'react-native'
import Login from '../components/Login'
import CustomButton from "../components/CustomButton"
import AuthLayout from '../components/AuthLayout'
import Register from '../components/Register/Register'

```




```

import { useDispatch } from 'react-redux'
import { getAllCircular } from '../actions/circular'
const Auth = ({ navigation }) => {
  const [login,setLogin]=useState(true)
  const dispatch=useDispatch()
  useEffect(()=>{
    dispatch(getAllCircular())
  })
  return (
    <AuthLayout>
    <View style={styles.btnWrapper}>
      <View style={styles.btn}>
        <CustomButton text='Log In'
onPress={()=>setLogin(true)} type={login?'selected':'auth'}/>
      </View>
      <View style={styles.btn}>
        <CustomButton text='Sign Up'
onPress={()=>setLogin(false)} type={!login?'selected':'auth'}/>
      </View>
    </View>
    <View style={styles.form}>
      {login?
        <Login navigation={navigation} />:
        <Register navigation={navigation} />
      }
    </View>
  </AuthLayout>
)
}
const styles=StyleSheet.create({
  btnWrapper:{

```



```

    flexDirection:'row',
    backgroundColor:"#F9F7F7",
    borderRadius:20
  },
  form:{
    marginVertical:20,
  },
  btn:{
    flex:1
  }
}
export default Auth

```

Circular.js

```

import { View, Text,StyleSheet, FlatList,SectionList, ActivityIndicator } from 'react-native'
import React, { useEffect, useState } from 'react'
import { useDispatch,useSelector } from 'react-redux'
import CustomButton from '../components/CustomButton'
import CircularCard from '../components/CircularCard'
import { getAllCircular } from '../actions/circular'
import Icon from 'react-native-vector-icons/MaterialIcons'
import AsyncStorage from '@react-native-async-storage/async-storage'
import { deleteDeviceId } from '../actions/auth'
const Item = ({ data,navigation }) => {
  return(
    <View style={styles.item}>
      <CircularCard item={data} navigation={navigation} />
    </View>
  )
}

```



```

const Circular = ({navigation}) => {
  const [dis,setDis]=useState(false)
  const [today,isToday]=useState(true)
  const [yesterday,isYesterday]=useState(false)
  const [earlier,isEarlier]=useState(false)
  const dispatch=useDispatch()
  const Circulars = useSelector((state) =>(state.circularReducer))
  const [todayCircular,setTodayCircular] = useState()
  const [yesterdayCircular,setYesterdayCircular] = useState()
  const [ earlierCircular,setEarlierCircular] = useState()
  const [monthWiseCircular,setMonthWiseCircular]=useState()
  const User = useSelector((state)=>(state.currentUserReducer))
  const setToday = () =>{
    isToday(true)
    isYesterday(false)
    isEarlier(false)
  }
  const setYesterday = () =>{
    isToday(false)
    isYesterday(true)
    isEarlier(false)
  }
  const setEarlier = () =>{
    isToday(false)
    isYesterday(false)
    isEarlier(true)
  }
  const handleLogout =async()=>{
    const id=User?.result?._id
    console.log(id)
  }
}

```



```

    dispatch(deleteDeviceId({id},navigation))
  }
  useEffect(()=>{
    if(Circulars.data!=null){
      const data = Circulars.data
      setTodayCircular(data.todayCircular)
      setYesterdayCircular(data.yesterdayCircular)
      setEarlierCircular(data.allCircular)
      const month = data.monthwise
      let fullData=[];
      for(let i of month){

        if(i.data.length){
          fullData.push(i)
        }
      }
      setMonthWiseCircular(fullData)
      setDis(true)
    }
  },[Circulars])
  useEffect(()=>{
    dispatch(getAllCircular())
  },[dispatch])
  return (
    <View style={styles.container}>
      <View style={styles.header}>
        <View style={styles.profile}>
          <Icon name="account-circle" color="#3F72AF" size={30}/>
          <Text style={styles.title}>{User?.result?.name}</Text>
        </View>

```



```

<View>
  <CustomButton type='logout' onPress={handleLogout} text='Logout' />
</View>
</View>
<View style={styles.buttons}>
  <CustomButton text='Today' onPress={setToday} type={today ? 'selected': 'normal'}
/>
  <CustomButton text='Yesterday' onPress={setYesterday} type={yesterday
?'selected': 'normal'} />
  <CustomButton text='Earlier' onPress={setEarlier} type={earlier ? 'selected': 'normal'}
/>
</View>
<View>
  { !dis && <ActivityIndicator /> }
  {today &&
    <FlatList
      data={todayCircular}
      keyExtractor={item => item._id}
      renderItem={({item}) => <CircularCard item={item} navigation={navigation} />}
    />
  }
  {yesterday &&
    <FlatList
      data={yesterdayCircular}
      keyExtractor={item => item._id}
      renderItem={({item}) => <CircularCard item={item} navigation={navigation}
/>}
    />
  }
  {earlier &&
    <SectionList
      sections={monthWiseCircular}

```



```

    keyExtractor={(item, index) => item + index}
    renderItem={({ item }) => <Item data={item} navigation={navigation} />}
    renderSectionHeader={({ section: { title } }) => (
      <Text style={styles.Sectiontitle}>{title }</Text>
    )}
  />
</View>
</View>
)
}

```

```

const styles= StyleSheet.create({
  container:{
    flex:1
  },
  ctitle:{
    fontSize: 32,
    backgroundColor: "#fff"
  },
  header:{
    flexDirection:"row",
    justifyContent:"space-between",
    alignItems:"center",
    margin:10
  },
  buttons:{
    flexDirection:"row",
    justifyContent:"space-between",
    alignItems:"center",
    padding:14,

```



```
    },  
    title:{  
      // marginHorizontal:30,  
      // marginVertical:20,  
      fontSize:24,  
      fontWeight:'500',  
      color:"#112D4E",  
      marginLeft:5,  
      textTransform:'capitalize'  
    },  
    Sectiontitle:{  
      margin:10,  
      fontSize:28,  
      color:"#112D4E",  
      fontWeight:'bold'  
    },  
    profile:{  
      flexDirection:'row',  
      justifyContent:'center',  
      alignItems:"center",  
  
    }  
  })  
  export default Circular
```



OUTPUT:

Login

Email

Password

Login

SignUp

Name

Email

Password

Confirm Password

User Type

Admin

Department (if department admin)

None

SignUp





Today

Yesterday

Earlier

November

Circular Number: 24
Circular Title: Testing
Batch: all
Department: all

>

Nov 04 2022

Circular Number: 23
Circular Title: Demo
Batch: all
Department: all

>

Nov 04 2022

Circular Number: 2
Circular Title: Demo
Batch: all
Department: all

>

Nov 03 2022



New Academic year

Add Circular

Circular No.

Title

Select Batch

Select Departments

All
2026
2025
2024
2023

All
Civil Engineering
Mechanical Engineering
Mechatronics Engineering
Automobile Engineering

Send To

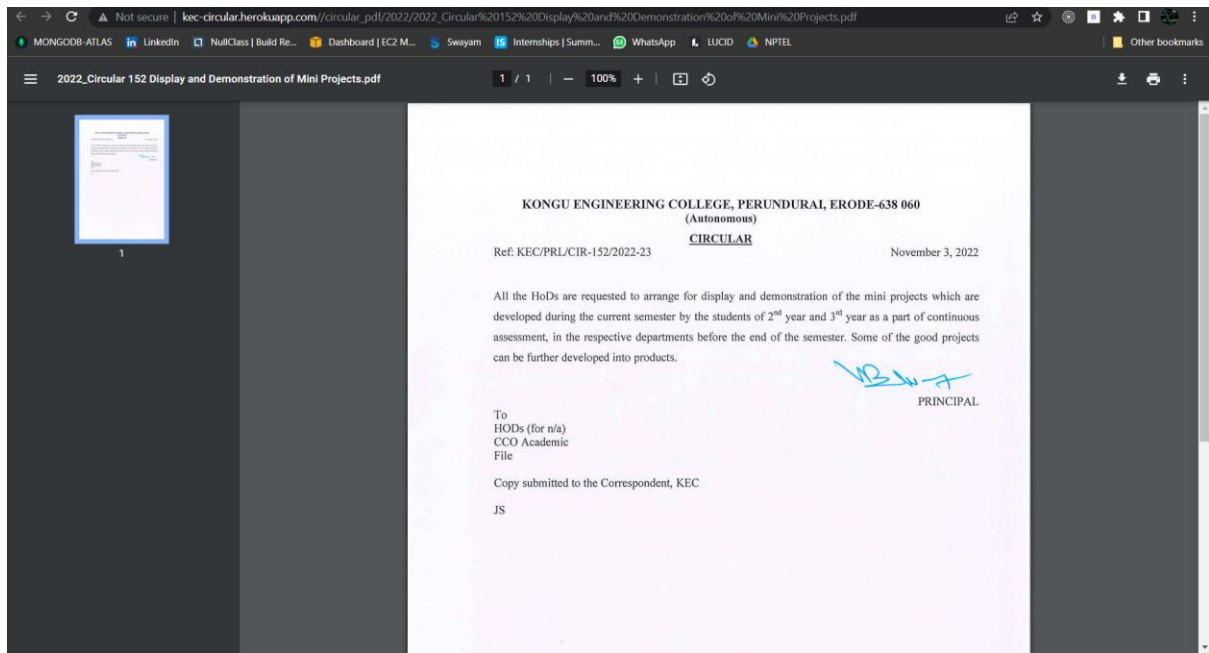
Select Circular File(Only pdf)

All


Choose file No file chosen

Add Circular






8:27 PM 4G+ 60%




Email ID

Password 


[forgot password?](#)


8:27 PM 4G+ 60%




Name

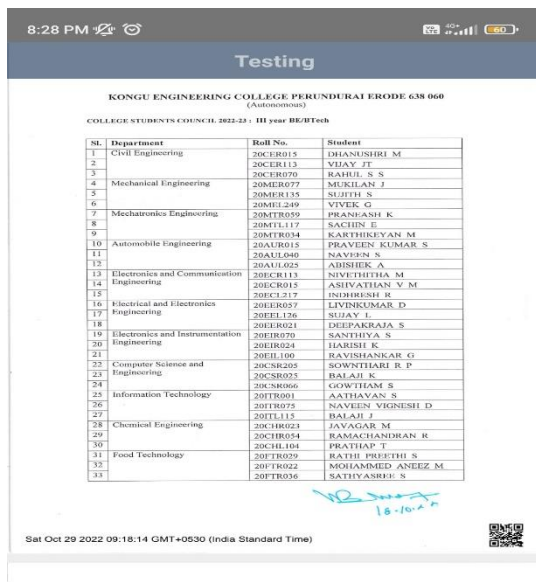
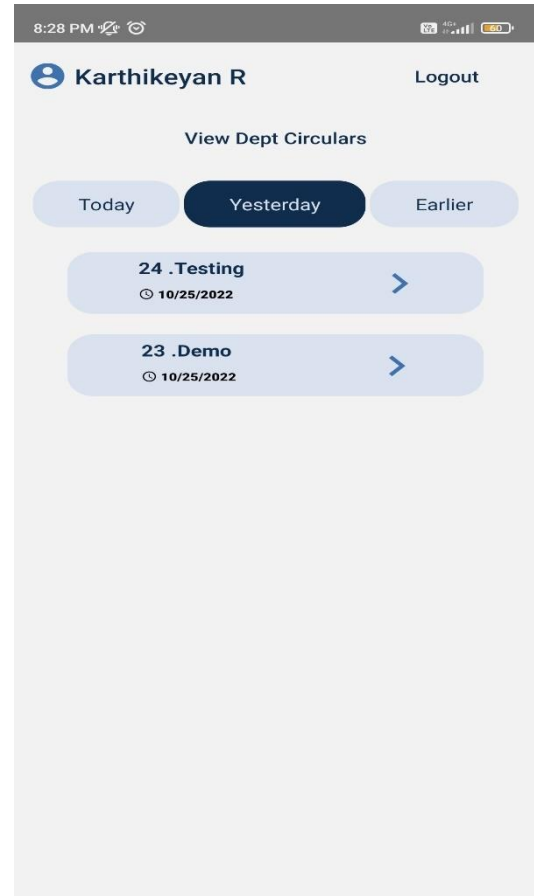
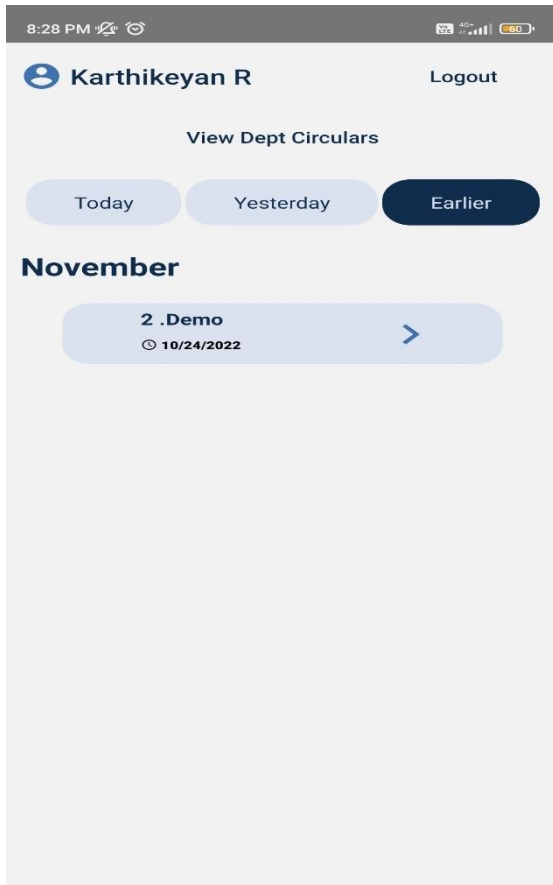
Email ID

Select Department 

Password 

Confirm Password 





KONGU ENGINEERING COLLEGE PERUNDURAI ERODE 638 060
(Autonomous)

COLLEGE STUDENTS COUNCIL 2022-23 - II year MCA

Sl.	Department	Roll No.	Student
1	Master of Computer Applications	21MCR072	NIRUBA D
	Master of Computer Applications	21MCR054	LOKESH P
	Master of Computer Applications	21MCR033	JAGAN D



HOSTING ADDRESS:

<https://kec-circular.herokuapp.com/>

Result:

Thus the KEC Circular web and mobile application was created and hosted in online hosting platform successfully

