Web Programming Project Proposal

<Homework Calendar>

01286233 Web Programming

Software Engineering Program

By

65011430 Parisorn Prasartkul

65011445 Peeranat Leelawattanapanit

65011544 Sorawis Chongterdtoonskul

Web Programming Project

<Homework Calendar>

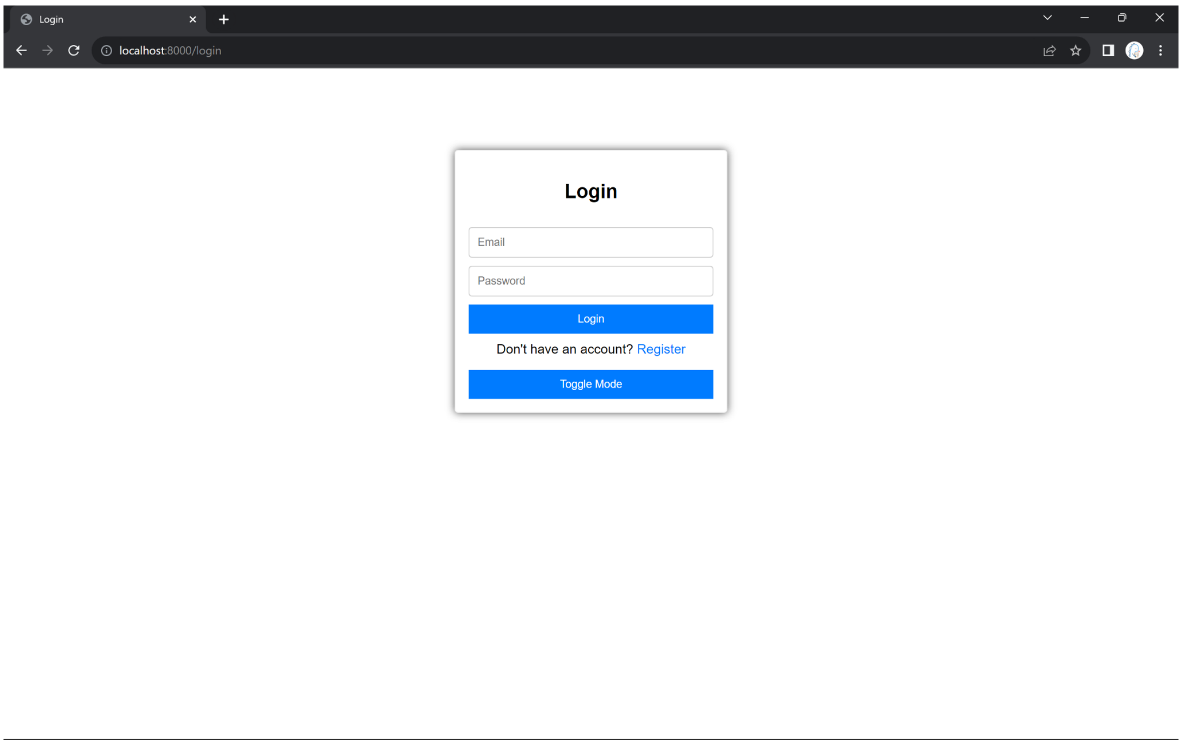
Project Description

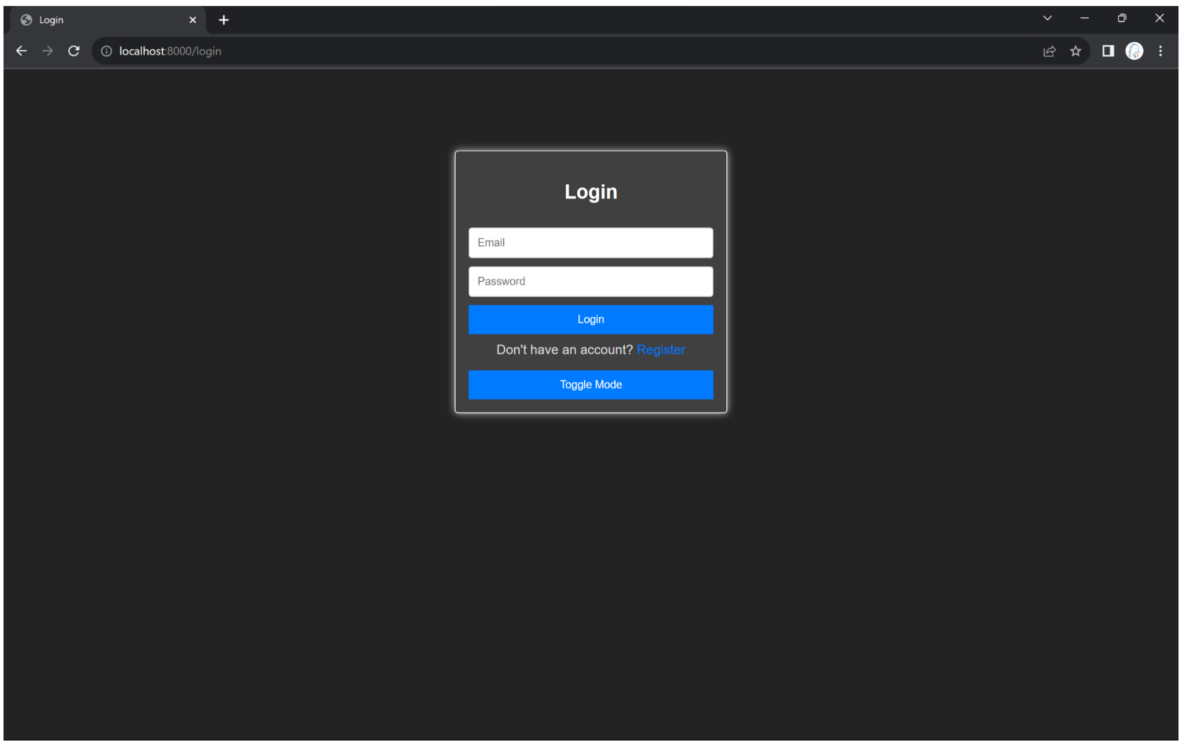
Homework Calendar is a web application for SE students to keep track of assignments. There is a login page as well as a registration page. Students can create an account and log in with their KMITL email address and a password. There is a separate login page for admins, accessible to users who know the admin passcode. The admin page allows authorized users to add, edit, and remove student accounts, assignments, and important events in each month. Admins can also give specific users the ability to add assignments and events to the site. Allowed users are the only users that could post an assignment or events into the calendar. Assignment information includes assignment name, course name, due date, and details or instructions. Once an assignment is added, it will appear in the main calendar for all students to view details and discuss. Students can use the forum feature under each assignment post to communicate. Each message in the forum will have information such as the email of the student and time of comment, and it also supports replying to previous users' comments. Students can also upload and download files.

HTML, CSS and Javascript were used for the front-end user interface. FastAPI is used extensively to get and post information onto the web, such as getting existing assignments and assignment details, and posting for adding new assignments from HTML form data, etc. For the database, ZODB is used to store students' log in data, assignment information, and forum messages. Jinja2 templates are used to pass objects and variables into HTML files dynamically. The web application is also responsive and users can toggle between light and dark themes.

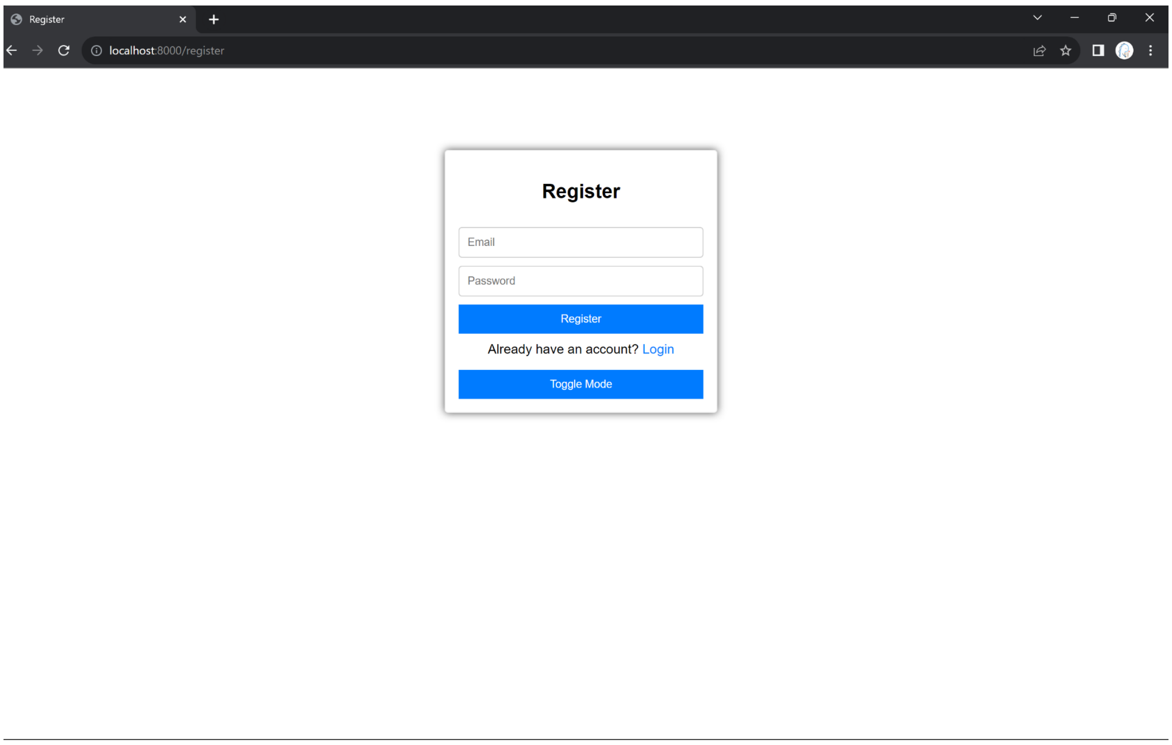
This project can be used in the SE website as a resource for students to view their upcoming assignments. It will also allow for productive discussions about assignment information, which will benefit student's learning and provide a platform to help connect the SE community.

Screen Captures





Login page



Register

localhost:8000/register

Register

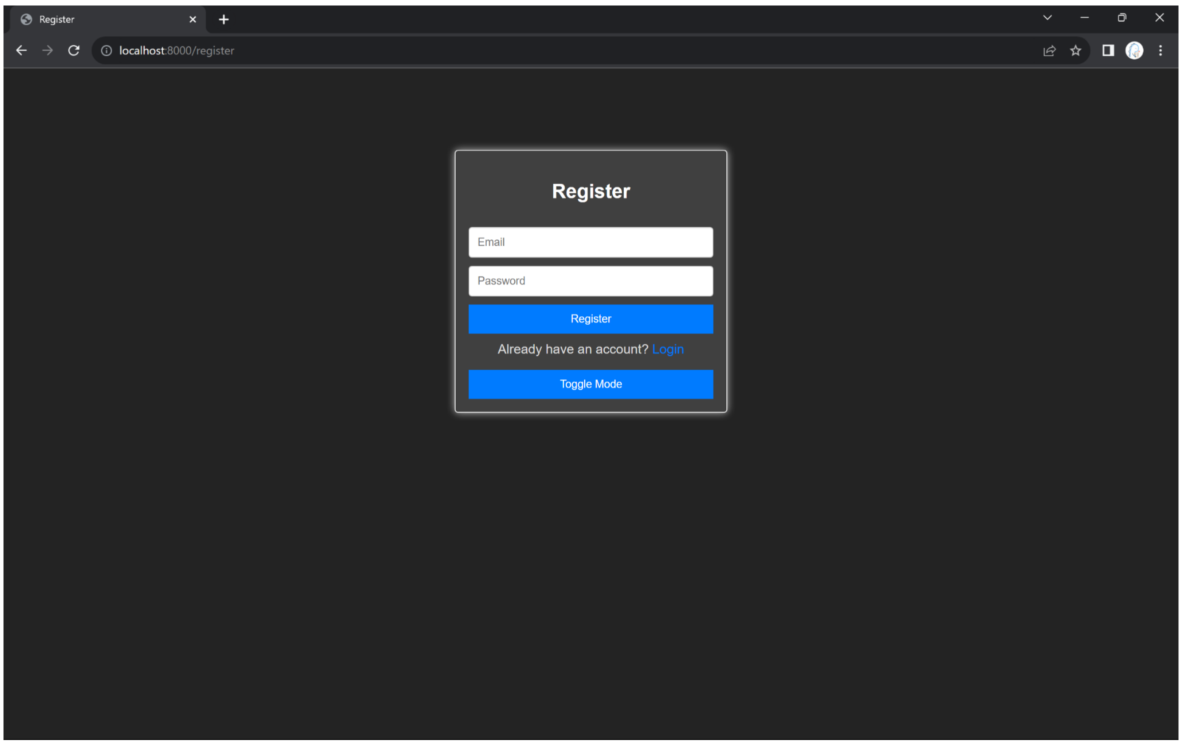
Email

Password

Register

Already have an account? Login

Toggle Mode



Register

localhost:8000/register

Register

Email

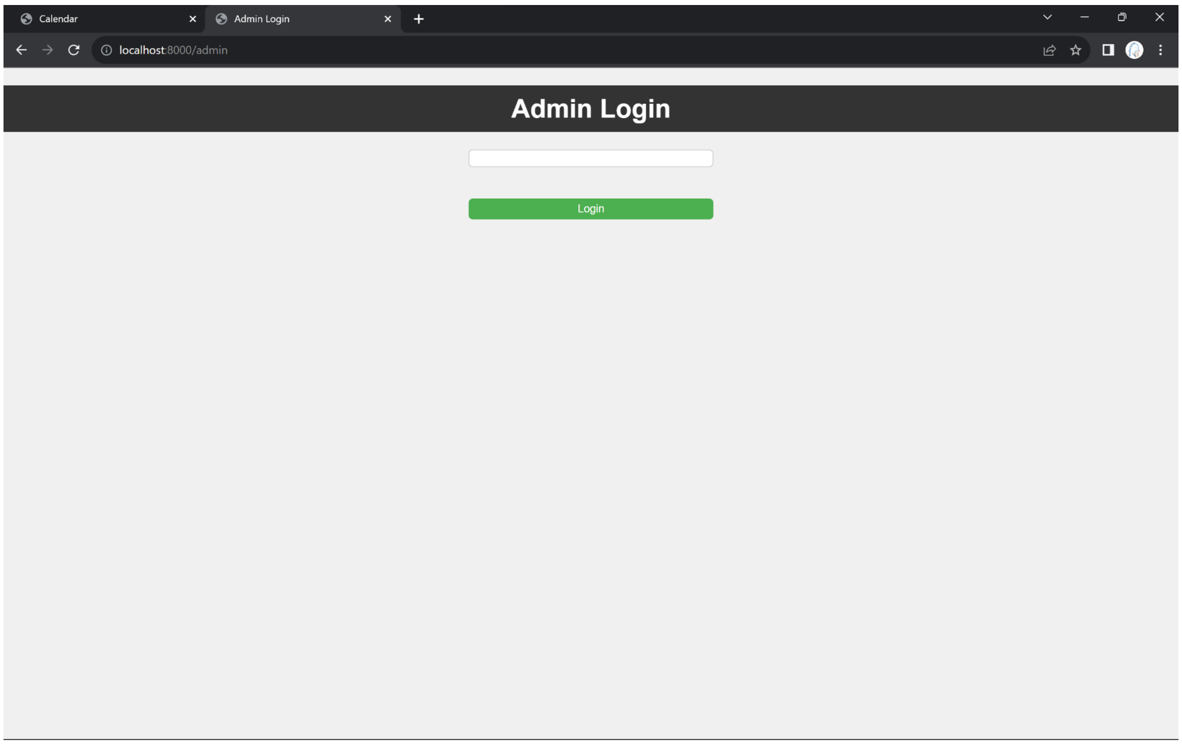
Password

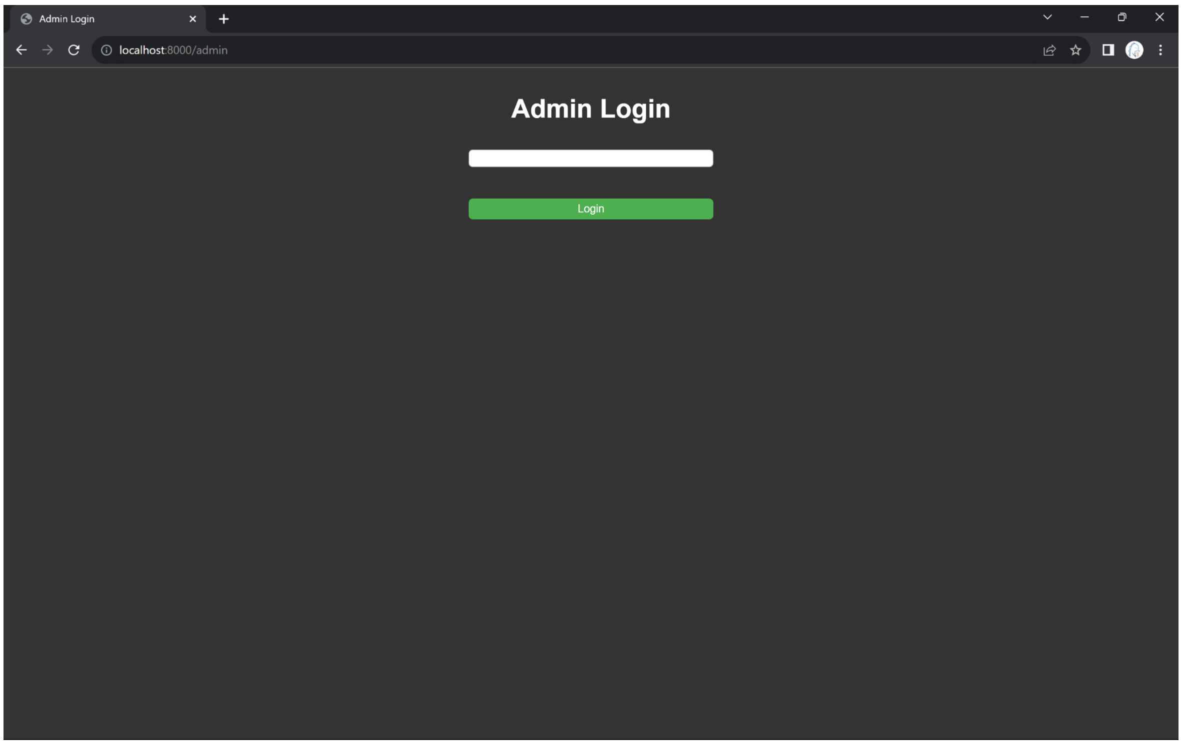
Register

Already have an account? Login

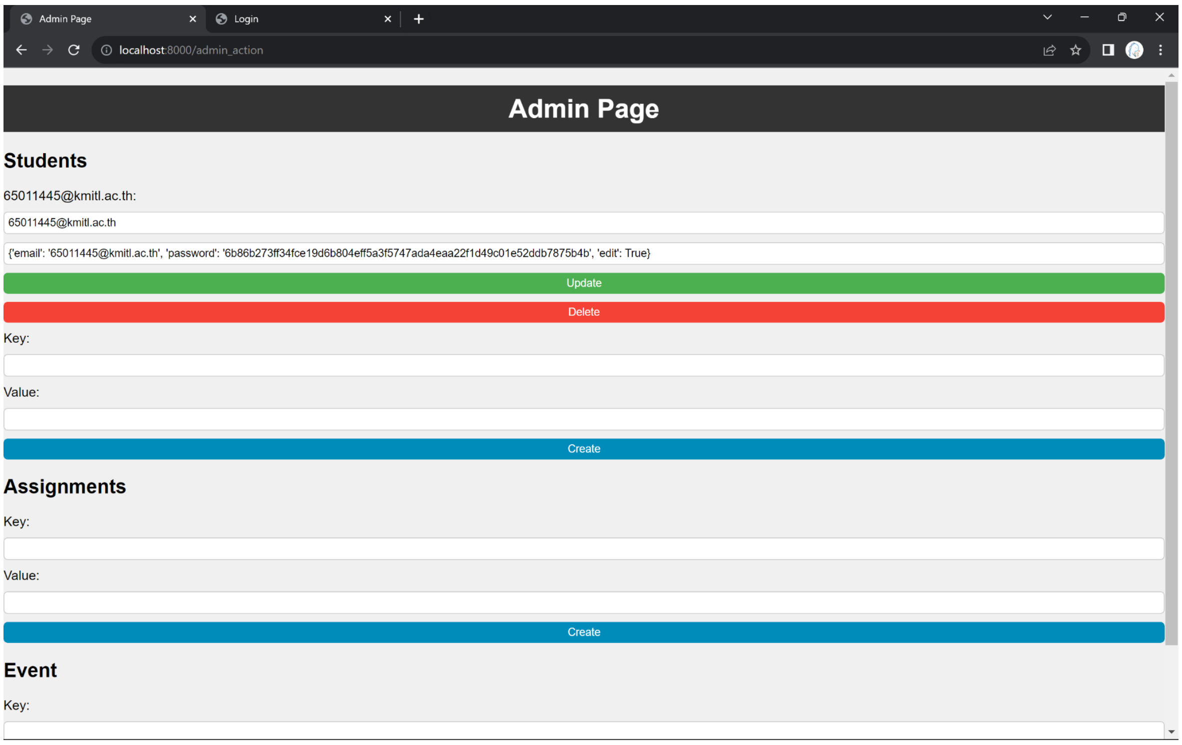
Toggle Mode

Register Page





Admin Login Page



Admin Page

Students

65011445@kmitl.ac.th:

65011445@kmitl.ac.th

{'email': '65011445@kmitl.ac.th', 'password': '6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b', 'edit': True}

Update

Delete

Key:

Value:

Create

Assignments

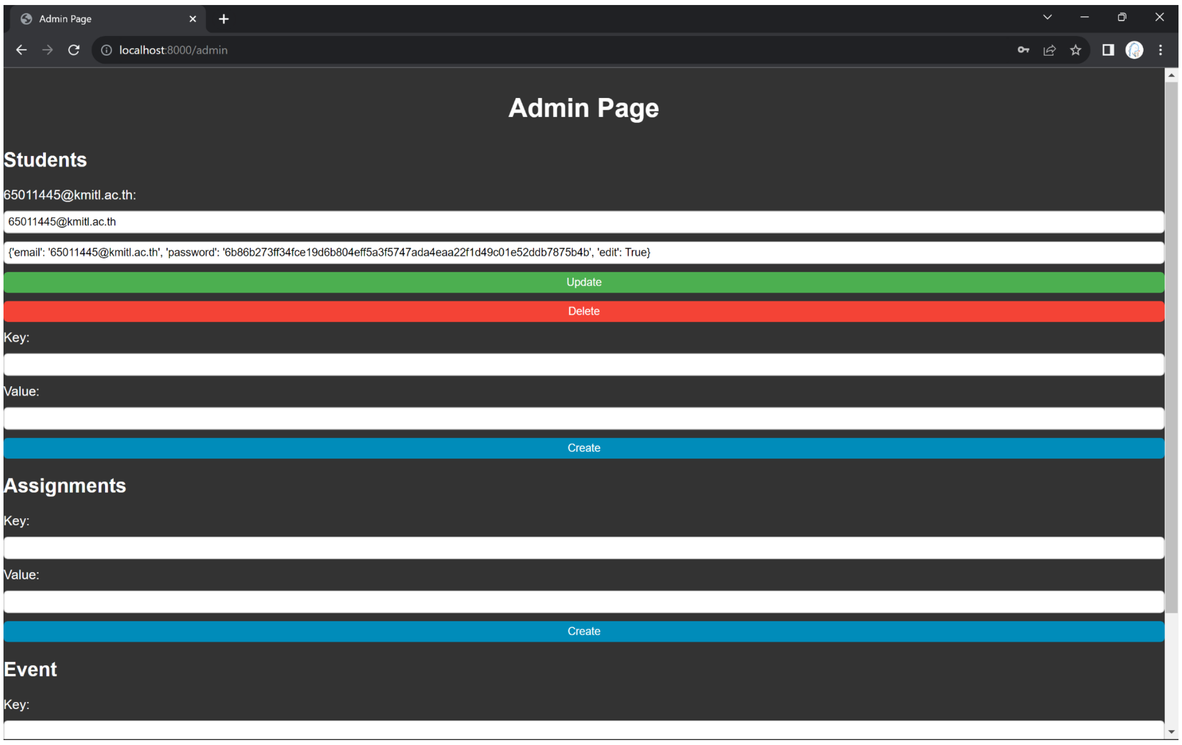
Key:

Value:

Create

Event

Key:



Admin Page

Students

65011445@kmitl.ac.th:

65011445@kmitl.ac.th

{'email': '65011445@kmitl.ac.th', 'password': '6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b', 'edit': True}

Update

Delete

Key:

Value:

Create

Assignments

Key:

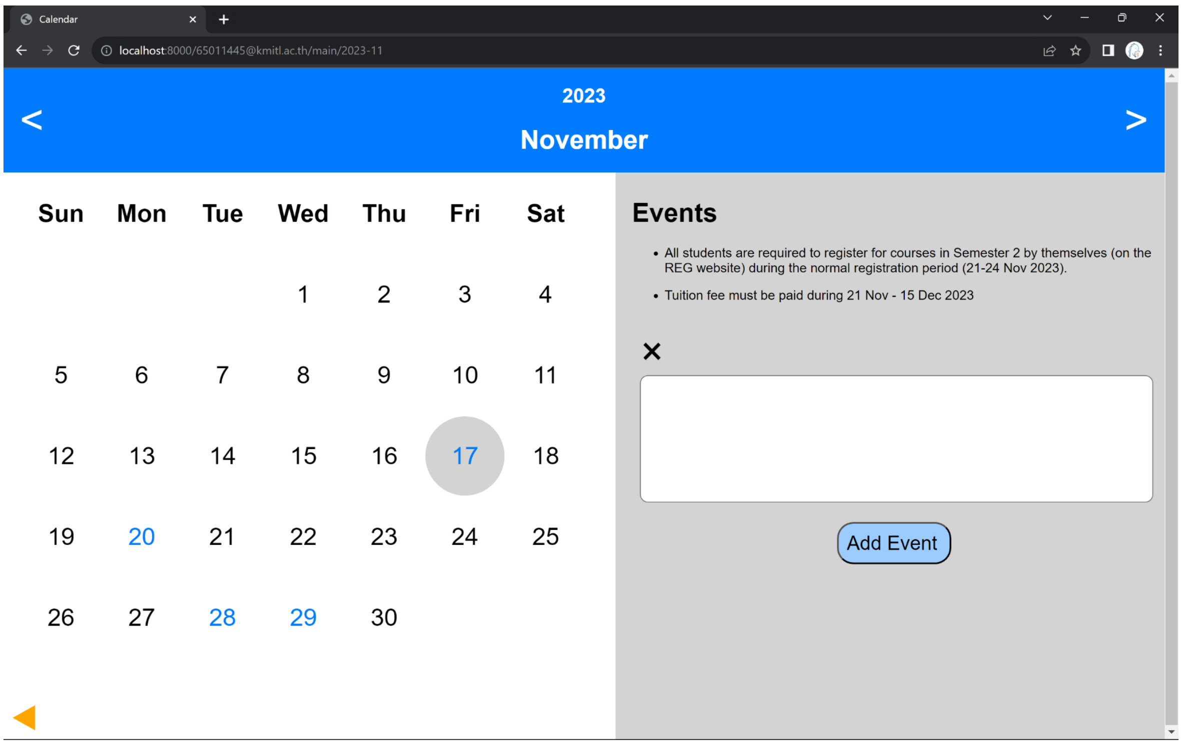
Value:

Create

Event

Key:

Admin Action Page to create, update, and delete student accounts, assignments and events. There is also a user log.



Calendar

localhost:8000/65011445@kmitl.ac.th/main/2023-11

2023

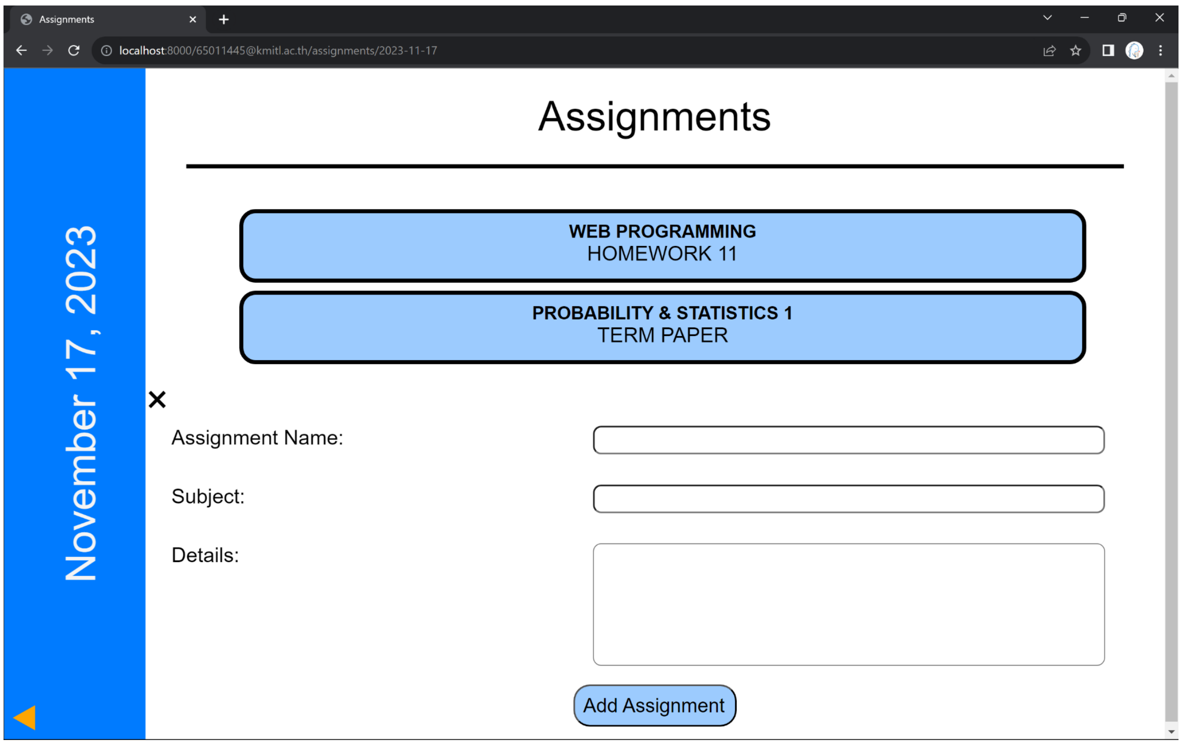
November

Sun	Mon	Tue	Wed	Thu	Fri	Sat
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

Events

- All students are required to register for courses in Semester 2 by themselves (on the REG website) during the normal registration period (21-24 Nov 2023).
- Tuition fee must be paid during 21 Nov - 15 Dec 2023

Main Page with interactive calendar (linking to assignment pages) and monthly events list. Allowed users also have an event creator.



Assignments

localhost:8000/65011445@kmitl.ac.th/assignments/2023-11-17

November 17, 2023

Assignments

WEB PROGRAMMING
HOMEWORK 11

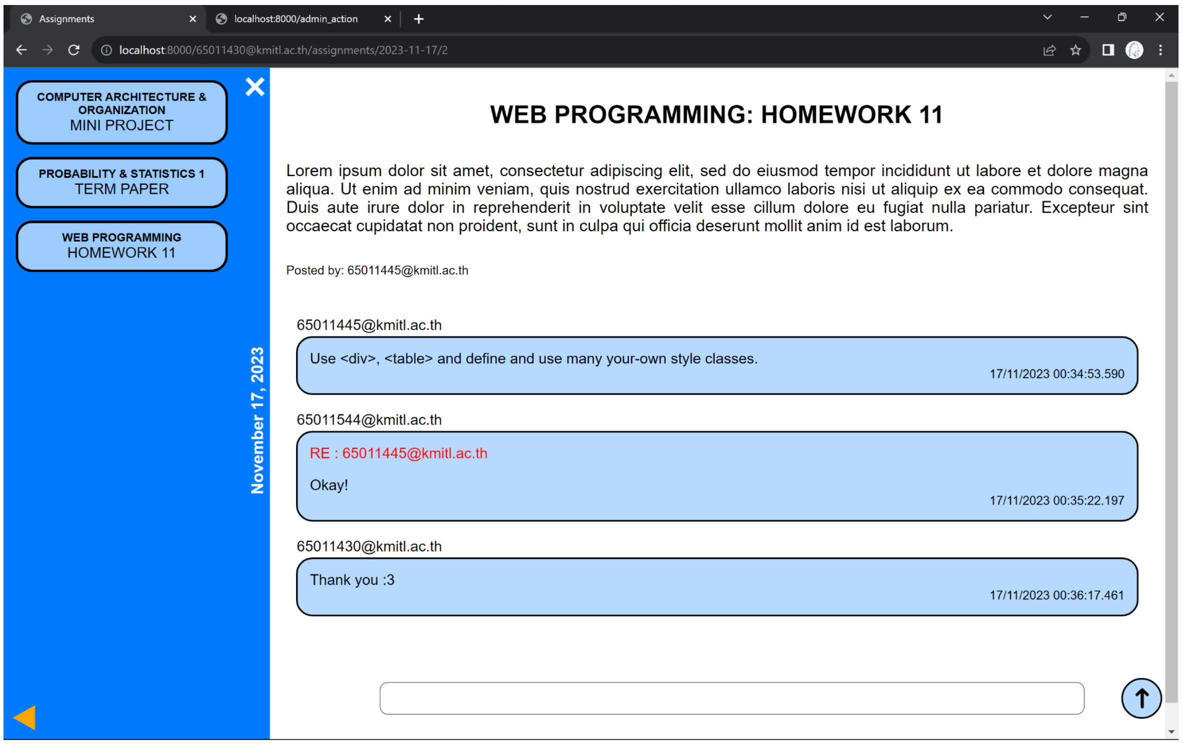
PROBABILITY & STATISTICS 1
TERM PAPER

Assignment Name:

Subject:

Details:

Assignment Page with assignment list and creator (for allowed users)


Assignments
localhost:8000/admin_action
localhost:8000/65011430@kmitl.ac.th/assignments/2023-11-17/2
COMPUTER ARCHITECTURE & ORGANIZATION
MINI PROJECT
PROBABILITY & STATISTICS 1
TERM PAPER
WEB PROGRAMMING
HOMEWORK 11
November 17, 2023

WEB PROGRAMMING: HOMEWORK 11

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Posted by: 65011445@kmitl.ac.th

65011445@kmitl.ac.th

Use <div>, <table> and define and use many your-own style classes. 17/11/2023 00:34:53.590

65011544@kmitl.ac.th

RE : 65011445@kmitl.ac.th

Okay! 17/11/2023 00:35:22.197

65011430@kmitl.ac.th

Thank you :3 17/11/2023 00:36:17.461

Assignment details and forum page.

Source Code

API.py

```
from fastapi import FastAPI, Request, Form, HTTPException, Depends, Body, File, UploadFile

from fastapi.templating import Jinja2Templates

from fastapi.responses import HTMLResponse, RedirectResponse, FileResponse

from fastapi.staticfiles import StaticFiles

from starlette.status import HTTP_303_SEE_OTHER

from datetime import datetime

import hashlib

import json

import ZODB, ZODB.FileStorage

from BTrees.OOBTree import BTree

import transaction

import persistent

from persistent.list import PersistentList

import re

import uuid

storage = ZODB.FileStorage.FileStorage('database.fs')

db = ZODB.DB(storage)

connection = db.open()

root = connection.root



class Assignment(persistent.Persistent):

    def __init__(self, name, subject, due_date, content, posted_by):

        self.name = name

        self.subject = subject

        self.due_date = due_date
        self.content = content
        self.forum = BTree()
        self.posted_by = posted_by

    def add_message(self, comment, user, reply_user=None, filename=None):
        time = datetime.now().strftime("%d/%m/%Y %H:%M:%S.%f")[:-3]
        new_message = Message(comment, user, time, reply_user, filename)
        self.forum[time] = new_message



class Message(persistent.Persistent):
    def __init__(self, comment, user, time, reply_user=None, filename=None):
        self.reply_user = reply_user
        self.comment = comment
        self.user = user
        self.time = time
        self.filename = filename



class Student(persistent.Persistent):
    def __init__(self, email, password, edit):
        self.email = email
        self.password = password
        self.edit = edit

class Event(persistent.Persistent):
    def __init__(self, yyyymm, message):
        self.yyyymm = yyyymm
        self.events = PersistentList([message])
```

```python
class LoginHistory(persistent.Persistent):
    def __init__(self, email, ip_address):
        self.time = datetime.now().strftime("%d/%m/%Y %H:%M:%S")
        self.email = email
        #self.password = password
        self.ip_address = ip_address


if not hasattr(root, 'students'):
    root.students = BTree()
if not hasattr(root, 'assignments'):
    root.assignments = BTree()
if not hasattr(root, 'events'):
    root.events = BTree()
if not hasattr(root, 'login_history'):
    root.login_history = BTree()
if not hasattr(root, 'visual'):
    root.visual = {}


app = FastAPI()

templates = Jinja2Templates(directory="templates")
app.mount("/static", StaticFiles(directory="static"), name="static")

def hash_password(password: str):
    return hashlib.sha256(password.encode()).hexdigest()
```

```
@app.get("/register", response_class=HTMLResponse)

async def register(request: Request):

    if request.client.host not in root.visual:

        root.visual[request.client.host] = "dark_mode"

        transaction.commit()

         return  templates.TemplateResponse("register.html",  {"request":  request  ,  "visual":
root.visual[request.client.host]})



@app.post("/register", response_class=HTMLResponse)

async def register(request: Request, email: str = Form(...), password: str = Form(...)):

    students = root.students

    if email in students:

        raise HTTPException(status_code=400, detail="Email already registered")

    if not re.match(r"[0-9]+@kmitl\.ac\.th", email):

        raise HTTPException(status_code=400, detail="Invalid email format")

    hashed_password = hash_password(password)

    new_student = Student(email, hashed_password, False)

    students[email] = new_student

    transaction.commit()

    return RedirectResponse(url=f"/login", status_code=HTTP_303_SEE_OTHER)



@app.get("/login", response_class=HTMLResponse)

async def login(request: Request):

    if request.client.host not in root.visual:

        root.visual[request.client.host] = "dark_mode"

        transaction.commit()

            return  templates.TemplateResponse("login.html",  {"request":  request,  "visual":
root.visual[request.client.host]})



@app.post("/login", response_class=HTMLResponse)
```

```python
async def login(request: Request, email: str = Form(...), password: str = Form(...)):
    students = root.students
    if email in students and students[email].password == hash_password(password):
        yyyymm = datetime.now().strftime("%Y-%m")
        ip_address = request.client.host
        root.login_history[email] = LoginHistory(email, ip_address)
        transaction.commit()
        return RedirectResponse(url=f"/{email}/main/{yyyymm}", status_code=HTTP_303_SEE_OTHER)
      return  templates.TemplateResponse("error.html",  {"request":  request,  "error":  "Incorrect login"})


def is_logged_in(email: str = None):
    if email not in root.students:
       raise HTTPException(status_code=HTTP_303_SEE_OTHER, detail="/login")
    return root.students[email].email


@app.get("/{email}/main/{yyyymm}", response_class=HTMLResponse)
async def get_by_month(request: Request, yyyymm: str, email: str = Depends(is_logged_in)):
    can_edit : bool = root.students[email].edit
    if yyyymm in root.events:
        events = root.events[yyyymm].events
    else:
        events = []


    assignment_days = []
    for date, assignments_list in root.assignments.items():
        assignment_yyyymm = date[:7]
        if (assignment_yyyymm == yyyymm and assignments_list):
            assignment_days.append(int(date[-2:]))
    return templates.TemplateResponse("main.html", {"request": request, "email": email, "events":
events, "yyyymm": yyyymm, "assignment_days": assignment_days, "can_edit": can_edit, "visual":
root.visual[request.client.host]})


@app.post("/{email}/main/{yyyymm}", response_class=HTMLResponse)
async def add_event(request: Request, yyyymm: str, content: str = Form(...), email: str =
Depends(is_logged_in)):
    if yyyymm in root.events:
        root.events[yyyymm].events.append(content)
    else:
        new_event = Event(yyyymm, content)
        root.events[yyyymm] = new_event;
    transaction.commit()
    return RedirectResponse(url=f"/{email}/main/{yyyymm}", status_code=HTTP_303_SEE_OTHER)


@app.get("/{email}/assignments/{date}", response_class=HTMLResponse)
async def get_assignments(request: Request, date: str, email: str = Depends(is_logged_in)):
    can_edit : bool = root.students[email].edit
    assignments = []
    if date in root.assignments:
        for assignment in root.assignments[date]:
            assignments.append({"subject": assignment.subject, "name": assignment.name})
    return templates.TemplateResponse("assignment.html", {"request": request, "email": email,
"assignments": assignments, "date": date, "can_edit": can_edit, "visual":
root.visual[request.client.host]})


@app.post("/{email}/assignments/{date}", response_class=HTMLResponse)
async def add_assignment(request: Request, date: str, assignment_name: str = Form(...), subject:
str = Form(...), content: str = Form(...), email: str = Depends(is_logged_in)):
    if date not in root.assignments:
        root.assignments[date] = PersistentList()
    new_assignment = Assignment(assignment_name, subject, date, content, email)
```

```python
    root.assignments[date].append(new_assignment)

    transaction.commit()

    return RedirectResponse(url=f"/{email}/assignments/{date}", status_code=HTTP_303_SEE_OTHER)



@app.get("/{email}/assignments/{date}/{assignment_index}", response_class=HTMLResponse)

async def get_assignment(request: Request, date: str, assignment_index: int, email: str = Depends(is_logged_in)):

    assignments = []

    if date in root.assignments:

        for assignment in root.assignments[date]:

            assignments.append({"subject": assignment.subject, "name": assignment.name})

    assignment_obj = root.assignments[date][assignment_index]

    return templates.TemplateResponse("forum.html", {"request": request, "email": email, "assignments": assignments, "date": date, "assignment": assignment_obj, "visual" : root.visual[request.client.host]})



@app.post("/{email}/assignments/{date}/{assignment_index}", response_class=HTMLResponse)

async def add_forum_msg(request: Request, date: str, assignment_index: int, email: str = Depends(is_logged_in), reply_user: str = Form(None), comment: str = Form(...), file: UploadFile = File(None)):

    filename = None

    if file.size > 0:

        filename = str(uuid.uuid4()) + file.filename

        unique_filename = "./static/" + filename

        with open(unique_filename, "wb") as buffer:

            buffer.write(await file.read())


    assignment_obj = root.assignments[date][assignment_index]

    assignment_obj.add_message(comment, email, reply_user, filename)

    transaction.commit()

    return RedirectResponse(url=f"/{email}/assignments/{date}/{assignment_index}", status_code=HTTP_303_SEE_OTHER)
```

```python
@app.post("/visual")
async def visual(request: Request):
    ip_address = request.client.host
    if root.visual[ip_address] == "light_mode":
        root.visual[ip_address] = "dark_mode"
    else:
        root.visual[ip_address] = "light_mode"
    transaction.commit()
    return RedirectResponse(url=f"/register", status_code=HTTP_303_SEE_OTHER)


@app.get("/logout")
async def logout(request: Request):
    ip_address = request.client.host
    del root.visual[ip_address]
    transaction.commit()
    return RedirectResponse(url=f"/login", status_code=HTTP_303_SEE_OTHER)


@app.delete("/delete_event/{yyyymm}/{index}")
async def delete_event(request: Request, yyyymm: str, index: int):
    del root.events[yyyymm].events[index]
    transaction.commit()


@app.delete("/delete_assignment/{date}/{index}")
async def delete_assignment(request: Request, date: str, index: int):
    del root.assignments[date][index]
    transaction.commit()


@app.get("/purge_database")
```

```
async def purge_database(request: Request):

    root.students = BTree()

    root.assignments = BTree()

    root.events = BTree()

    root.login_history = BTree()

    transaction.commit()

    return RedirectResponse(url="/admin", status_code=HTTP_303_SEE_OTHER)



@app.get("/admin", response_class=HTMLResponse)

async def admin_login(request: Request):

    if request.client.host not in root.visual:

        root.visual[request.client.host] = "dark_mode"

        transaction.commit()

         return  templates.TemplateResponse("admin_login.html",  {"request":  request,  "visual":
root.visual[request.client.host]})



@app.post("/admin", response_class=HTMLResponse)

async def admin_login(request: Request, password: str = Form(...)):

    if password == "admin":

        students = {email: student.__dict__ for email, student in root.students.items()}

                    #   assignments   =   {name:   assignment.__dict__   for   name,   assignment   in
root.assignments.items()}

        # events = {yyyymm: event.__dict__ for yyyymm, event in root.events.items()}

        logs = {email: log.__dict__ for email, log in root.login_history.items()}



        data = {

            "Students": students,

            # "Assignments": assignments,

            # "Event" : events,

            "Logs": logs
        }

        return templates.TemplateResponse("admin_page.html", {"request": request, "data": data ,
"visual": root.visual[request.client.host]})

    else:

        return templates.TemplateResponse("error.html", {"request": request, "error": "Incorrect
password"})



@app.post("/admin/log", response_class=HTMLResponse)

async def admin_log(request: Request, email: str = Form(...)):

    if email not in root.students:

        return templates.TemplateResponse("error.html", {"request": request, "error": "Invalid
email"})

    logs = [log.__dict__ for log in root.login_history if log.email == email]

    return templates.TemplateResponse("admin_log.html", {"request": request, "logs": logs})



@app.post("/admin_action", response_class=HTMLResponse)

async def admin_action(request: Request, action: str = Form(...), key: str = Form(...), value:
str = Form(...)):

    if action == "Delete":

        del root.students[key]

    elif action == "Update":

        args : dict = eval(value)

        root.students[args['email']] = Student(args['email'], args['password'], args['edit'])

    elif action == "Create":

        args : dict = eval(value)

        root.students[args['email']] = Student(args['email'], args['password'], args['edit'])

    else:

        return templates.TemplateResponse("error.html", {"request": request, "error": "Invalid
action"})

    transaction.commit()
    students = {email: student.__dict__ for email, student in root.students.items()}

    data = {

        "Students": students

    }

        return  templates.TemplateResponse("admin_page.html",  {"request":  request,  "data":  data,
"visual": root.visual[request.client.host]})
```

Register.html

```
<!DOCTYPE html>

<html>

<head>

    <title>Register</title>

    <style>

        body {

            font-family: Arial, sans-serif;

            background-color: rgb(35, 35, 35);

            margin: 0;

            padding: 0;

            transition: background-color 0.5s ease;

        }



        .container {

            max-width: 300px;

            padding: 16px;

            background-color: rgb(64, 64, 64);

            margin: 0 auto;
            margin-top: 100px;

            border: 1px solid #f4f4f4;

            border-radius: 4px;

            box-shadow: 0 0 10px rgba(255, 255, 255, 0.684);

            animation: slide-up 0.5s ease;

        }


        .container h2 {

            text-align: center;

            color: #dbdbdb;

            margin-bottom: 30px;

        }


        .container input {

            width: 100%;

            padding: 10px;

            margin-bottom: 10px;

            border-radius: 4px;

            border: 1px solid #ccc;

            box-sizing: border-box;

            transition: border-color 0.3s ease;

        }


        .container input:focus {

            border-color: #007BFF;

        }


        .container button {

            width: 100%;
```

```css
    padding: 10px;

    background-color: #007BFF;

    border: 0;

    color: white;

    cursor: pointer;

    transition: background-color 0.3s ease;

}



.container button:hover {

    background-color: #0056b3;

}



.container p {

    text-align: center;

    margin-top: 10px;

    color: #dbdbdb;

}



.container p a {

    color: #007BFF;

    text-decoration: none;

    transition: color 0.3s ease;

}



.container p a:hover {

    text-decoration: underline;

    color: #84bfff;

}
```

```
    @media screen and (max-width: 600px) {

        .container {

            margin-top: 20px;

            width: 80%;

        }

    }



    @keyframes slide-up {

        0% {

            transform: translateY(50px);

            opacity: 0;

        }

        100% {

            transform: translateY(0);

            opacity: 1;

        }

    }

</style>

<script>

    function validateEmail(event) {

        var email = document.querySelector('input[name="email"]').value;

        if (!email.endsWith('@kmitl.ac.th')) {

            alert('Please use an email that ends with @kmitl.ac.th');

            event.preventDefault();

        }

        if (email.split('@')[0].length != 8 || isNaN(email.split('@')[0])) {

            alert('Email must be a valid ID!');

            event.preventDefault();

        }
        }



        function toggleMode() {

        fetch('/visual/', { method: 'POST' })

            .then(() => location.reload());

        }



        window.addEventListener('load', (event) => {

            on_load();

        });



        function on_load() {

                    if ("{{ visual }}" == 'dark_mode') {

                        document.body.style.backgroundColor = 'rgb(35, 35, 35)';

                         document.querySelector('.container').style.backgroundColor = 'rgb(64, 64,
64)';

                                 document.querySelector('.container').style.boxShadow = '0 0 10px
rgba(255, 255, 255, 0.684)';

                        document.querySelector('.container').style.borderColor = '#f4f4f4';

                        document.querySelector('.container').style.color = '#dbdbdb';

                        document.querySelector('.container h2').style.color = 'white';

                    } else {

                        document.body.style.backgroundColor = 'white';

                        document.querySelector('.container').style.backgroundColor = 'white';

                          document.querySelector('.container').style.boxShadow = '0 0 10px rgba(0,
0, 0, 0.684)';

                        document.querySelector('.container').style.borderColor = '#f4f4f4';

                        document.querySelector('.container').style.color = 'black';

                        document.querySelector('.container p').style.color = 'black';

                        document.querySelector('.container h2').style.color = 'black';
                    }
                }

    </script>
</head>
<body>
    <div class="container">
        <h2>Register</h2>
        <form action="/register" method="post" onsubmit="validateEmail(event)">
            <input type="email" name="email" placeholder="Email" required>
            <input type="password" name="password" placeholder="Password" required>
            <button type="submit">Register</button>
        </form>
        <p>Already have an account? <a href="/login">Login</a></p>
        <button onclick="toggleMode()">Toggle Mode</button>
    </div>
</body>
</html>
```

Login.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Login</title>
    <style>
```

```css
body {

    font-family: Arial, sans-serif;

    background-color: #f4f4f4;

    margin: 0;

    padding: 0;

    transition: background-color 0.5s ease;

}



.container {

    max-width: 300px;

    padding: 16px;

    background-color: white;

    margin: 0 auto;

    margin-top: 100px;

    border: 1px solid #f4f4f4;

    border-radius: 4px;

    box-shadow: 0 0 10px rgba(0,0,0,0.1);

    animation: slide-up 0.5s ease;

}



.container h2 {

    text-align: center;

    color: #333;

    margin-bottom: 30px;

}



.container input {

    width: 100%;

    padding: 10px;
    margin-bottom: 10px;

    border-radius: 4px;

    border: 1px solid #ccc;

    box-sizing: border-box;

    transition: border-color 0.3s ease;

}


.container input:focus {

    border-color: #007BFF;

}


.container button {

    width: 100%;

    padding: 10px;

    background-color: #007BFF;

    border: 0;

    color: white;

    cursor: pointer;

    transition: background-color 0.3s ease;

}


.container button:hover {

    background-color: #0056b3;

}


.container p {

    text-align: center;

    margin-top: 10px;

}
```

```
.container p a {

    color: #007BFF;

    text-decoration: none;

    transition: color 0.3s ease;

}


.container p a:hover {

    text-decoration: underline;

    color: #0056b3;

}


@media screen and (max-width: 600px) {

    .container {

        margin-top: 20px;

        width: 80%;

    }

}


@keyframes slide-up {

    0% {

        transform: translateY(50px);

        opacity: 0;

    }

    100% {

        transform: translateY(0);

        opacity: 1;

    }

}
```

```
    </style>
</head>
    <script>
        function validateEmail(event) {
            var email = document.querySelector('input[name="email"]').value;
            if (!email.endsWith('@kmitl.ac.th')) {
                alert('Please use an email that ends with @kmitl.ac.th');
                event.preventDefault();
            }
            if (email.split('@')[0].length != 8 || isNaN(email.split('@')[0])) {
                alert('Email must be a valid ID!');
                event.preventDefault();
            }
        }

        function toggleMode() {
        fetch('/visual/', { method: 'POST' })
            .then(() => location.reload());
        }

        window.addEventListener('load', (event) => {
            on_load();
        });

        function on_load() {
                    if ("{{ visual }}" == 'dark_mode') {
                        document.body.style.backgroundColor = 'rgb(35, 35, 35)';
                         document.querySelector('.container').style.backgroundColor = 'rgb(64, 64, 64)';
```

```
                                document.querySelector('.container').style.boxShadow = '0 0 10px
rgba(255, 255, 255, 0.684)';

                        document.querySelector('.container').style.borderColor = '#f4f4f4';

                        document.querySelector('.container').style.color = '#dbdbdb';

                        document.querySelector('.container h2').style.color = 'white';

                    } else {

                        document.body.style.backgroundColor = 'white';

                        document.querySelector('.container').style.backgroundColor = 'white';

                         document.querySelector('.container').style.boxShadow = '0 0 10px rgba(0,
0, 0, 0.684)';

                        document.querySelector('.container').style.borderColor = '#f4f4f4';

                        document.querySelector('.container').style.color = 'black';

                        document.querySelector('.container p').style.color = 'black';

                        document.querySelector('.container h2').style.color = 'black';

                    }

                }




    </script>

    <body>

    <div class="container">

        <h2>Login</h2>

        <form action="/login" method="post">

            <input type="email" name="email" placeholder="Email" required>

            <input type="password" name="password" placeholder="Password" required>

            <button type="submit">Login</button>

        </form>

        <p>Don't have an account? <a href="/register">Register</a></p>

        <button onclick="toggleMode()">Toggle Mode</button>

    </div>
```

```
</body>

</html>
```

main.html

```
<!DOCTYPE html>

<html>

    <head>

        <meta charset="utf-8">

        <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <title>Calendar</title>

        <style>

            body {

                font-family: Arial, sans-serif;

                margin: 0;

                padding: 0;

                display: flex;

                flex-direction: column;

                height: 100vh;

            }



            .flex-row {

                display: flex;

                flex-direction: row;

            }



            .header-box {

                flex: 1;

                background-color: #007BFF;
```

```css
        color: #ffffff;
        display: flex;
        text-align: center;
        justify-content: space-between;
    }

    button.navigation {
        background-color: #007BFF;
        color: #ffffff;
        font-size: 50px;
        border: 0;
        padding: 20px;
    }

    .calendar-box {
        flex: 7;
    }

    th, td {
        width: 95px;
        height: 95px;
        text-align: center;
        font-size: 30px;
        border-radius: 50%;
    }

    td.bubble { /* Indicates today. */
        background-color: #d3d3d3;
    }
```

```css
td.flag { /* Indicates there is an assignment on that date. */

    color: #007BFF;

}



table {

    padding-left: 20px;

}



.clickable {

    cursor: pointer;

}



.list-box {

    flex: 6;

    background-color: #d3d3d3;

    padding: 10px;

    padding-left: 20px;

    overflow: auto;

    word-wrap: break-word;

}



li {

    padding-bottom: 15px;

}



@media screen and (max-width: 800px) {

    .flex-row {

        display: flex;
```

```
                flex-direction: column;
            }
        }

        .modal {
            display: none;
            margin: 10px;
        }

        .close {
            display: flex;
            transition: .2s;
            cursor: pointer;
            font-size: 50px;
            width: fit-content;
            height: fit-content;
        }

        .close:hover{
            color: red;
        }

        #submit , #openModal{
            font-family: Arial, sans-serif;
            font-size: 24px;
            cursor: pointer;
            background-color: #9ccbfe;
            border-radius: 20px;
            display: flex;
```

```css
    margin-top: 50px;

    left: 50%;

    transition: .2s;

    width: 140px;

    margin: 20px auto;

    padding: 10px;

}


#submit:hover , #openModal:hover , .close:hover{

    transform: scale(1.1 , 1.1);

}


textarea{

    resize: none;

    font-size: 25px;

    width: 90%;

    border-radius: 10px;

    font-family: Arial, sans-serif;

    height: 150px;

    width: 100%;

}


select , input {

    font-size:  25px;

    font-family: Arial, sans-serif;

    width : 20% ;

    border-radius: 10px;

}
```

```
        #log-out {

            position: fixed;

            bottom: 0;

            left: 0;

            height: 50px;

            width: 50px;

            border: 0;

            background-color: transparent;

            font-size: 40px;

            color: orange;

            cursor: pointer;

        }


        #log-out:active {

            color: red;

        }


        @keyframes slideLeft {

            0% {transform: translateX(-800px);}

            100% {transform: translateX(0px);}

        }


        @keyframes slideRight {

            0% {transform: translateX(800px);}

            100% {transform: translateX(0px);}

        }


    </style>

</head>
```

```html
<body onload="on_load()">
    <div class="flex-row">
        <div class="header-box">
            <button class="navigation clickable" onclick="prevMonth()">&lt;</button>
            <div>
                <h2 id="year-header"></h2>
                <h1 id="month-header"></h1>
            </div>
            <button class="navigation clickable" onclick="nextMonth()">&gt;</button>
        </div>
    </div>

    <div class="flex-row">
        <div class="calendar-box">
            <table>
                <thead>
                    <tr>
                        <th>Sun</th>
                        <th>Mon</th>
                        <th>Tue</th>
                        <th>Wed</th>
                        <th>Thu</th>
                        <th>Fri</th>
                        <th>Sat</th>
                    </tr>
                </thead>
                <tbody id="calendar-body"></tbody>
            </table>
        </div>
```

```
            <div class="list-box">
                <h1>Events</h1>
                <ul id="list-items">
                    {% for event in events %}
                                <li>{{ event }} <input type="submit" value="Delete" name="action" onclick="resolveDelete('{{ loop.index0 }}')"></li>

                    {% endfor %}
                </ul>
                {% if can_edit == True %}
                    <button id="openModal">Add Event</button>
                    <div id="myModal" class="modal">
                        <div class="modal-content">
                            <div class="close">×</div>
                            <form onsubmit="return checkField()" method="post">
                                <textarea id="content" name="content"></textarea></td>
                                <input type="submit" value="Add Event" id="submit">
                            </form>
                        </div>
                    </div>
                {% endif %}
            </div>
        </div>
        <button id="log-out" onclick="logOut()">&#9664;</button>
        </body>
        <script>
            let now = new Date();
            let yyyymm = "{{ yyyymm }}";
```

```
    let shownYear = parseInt(yyyymm.slice(0, 4));
    let shownMonth = parseInt(yyyymm.slice(5, 7));
    let assignment_days = {{ assignment_days }};


    const yearHeader = document.getElementById("year-header");
    const monthHeader = document.getElementById("month-header");
    const calendar = document.getElementById("calendar-body");


    function generateCalendar(year, month) {
        month--; // Month index to 0 based.
        yearHeader.innerHTML = shownYear;
        monthHeader.innerHTML = new Date(shownYear, month).toLocaleDateString("en-US", {month: "long"});
        const firstDay = new Date(year, month, 1).getDay(); // Day is 0 based starting from Sunday.
        const lastDate = new Date(year, month + 1, 0).getDate();
        let date = 1;
        for (let week = 0; week < 6; week++) {
            const newWeek = document.createElement("tr");
            for (let day = 0; day < 7; day++) {
                const newDate = document.createElement("td");
                if ((week == 0 && day < firstDay) || date > lastDate) {
                    newDate.textContent = "";
                }
                else {
                    newDate.textContent = date.toString();
                    if (date == now.getDate() && month == now.getMonth() && year == now.getFullYear()) {
                        newDate.classList.add("bubble");
                    }
```

```
                            if (assignment_days.includes(date)) {
                                newDate.classList.add("flag");
                            }
                                    newDate.addEventListener("click", (function(date, shownMonth, shownYear) {
                                    return function() {
                                        let clickedDate = date < 10 ? `0${date}` : `${date}`;
                                            window.location.href = `/{{ email }}/assignments/${yyyymm}-${clickedDate}`;
                                    };
                                })(date, shownMonth, shownYear));
                            newDate.classList.add("clickable");
                            date++;
                        }
                        newWeek.appendChild(newDate);
                    }
                    calendar.appendChild(newWeek);
                }
            }
            generateCalendar(shownYear, shownMonth);

            function prevMonth() {
                if (shownMonth > 1) {
                    shownMonth--;
                }
                else {
                    shownMonth = 12;
                    shownYear--;
                }
                shownMonth = shownMonth < 10 ? `0${shownMonth}` : `${shownMonth}`;
                yyyymm = `${shownYear}-${shownMonth}`;
                let path = window.location.href.substring(0, window.location.href.lastIndexOf('/'));
                document.querySelector('body').style.animation = 'slideRight 0.8s ease-in-out';
                window.location.href = `/{{ email }}/main/${yyyymm}`;
            }

            function nextMonth() {
                if (shownMonth < 12) {
                    shownMonth++;
                }
                else {
                    shownMonth = 1;
                    shownYear++;
                }
                shownMonth = shownMonth < 10 ? `0${shownMonth}` : `${shownMonth}`;
                yyyymm = `${shownYear}-${shownMonth}`;
                let path = window.location.href.substring(0, window.location.href.lastIndexOf('/'));
                document.querySelector('body').style.animation = 'slideLeft 0.8s ease-in-out';
                window.location.href = `/{{ email }}/main/${yyyymm}`;
            }

            var modal = document.getElementById("myModal");
            var btn = document.getElementById("openModal");
            var span = document.getElementsByClassName("close")[0];

            btn.onclick = function() {
                modal.style.display = "block";
                btn.style.display = "none" ;
```

```
}

span.onclick = function() {
    modal.style.display = "none";
    btn.style.display = "block" ;
}

window.onclick = function(event) {
    if (event.target == modal) {
        modal.style.display = "none";
    }
}

function checkField() {
    let contentField = document.getElementById("content").value;
    if (contentField.trim() == "") {
        alert("Please fill out field to post.");
        return false;
    }
    return true;
}

function logOut() {
    window.location.href = `/logout`;
}

function resolveDelete(index)
{
fetch (`/delete_event/{{ yyyymm }}/${index}`, {method: 'DELETE'})
```

```
                .then(() => location.reload());

            }



            function on_load() {

                if ("{{ visual }}" == 'dark_mode') {

                    document.querySelector('body').style.backgroundColor = '#333';

                    document.querySelector('table').style.backgroundColor = '#333';

                    document.querySelector('table').style.color = '#f4f4f4';

                    document.querySelector('table').style.border = '5px solid #f4f4f4';

                    document.querySelector('table').style.borderRadius = '10px';

                        document.querySelector('table').style.boxShadow = '0 0 10px rgba(255, 255,
255, 0.684)';

                    document.querySelector('table').style.borderColor = '#f4f4f4';

                    document.querySelector('div.header-box').style.backgroundColor = '#333';

                    document.querySelector('div.header-box').style.color = '#f4f4f4';

                    document.querySelector('div.header-box').style.border = '5px solid #f4f4f4';

                    document.querySelector('div.header-box').style.borderRadius = '10px';

                            document.querySelector('div.header-box').style.boxShadow = '0 0 10px
rgba(255, 255, 255, 0.684)';

                    document.querySelector('div.header-box').style.borderColor = '#f4f4f4';

                    document.querySelector('div.list-box').style.backgroundColor = '#333';

                    document.querySelector('div.list-box').style.color = '#f4f4f4';

                    document.querySelector('div.list-box').style.border = '5px solid #f4f4f4';

                    document.querySelector('div.list-box').style.borderRadius = '10px';

                      document.querySelector('div.list-box').style.boxShadow = '0 0 10px rgba(255,
255, 255, 0.684)';

                    document.querySelector('div.list-box').style.borderColor = '#f4f4f4';



                }

            }
```

```
        </script>

</html>
```

Forum.html

```
<html>

    <head>

        <meta charset="utf-8">

        <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <title>Assignments</title>

        <style>

            body {

                font-family: Arial, sans-serif;

                margin: 0;

                padding: 0;

                display: flex;

            }



            div.sidebar{

                height: 100%;

                background-color: #007BFF;

                min-width: 290px;

                width : 290px;

                direction: rtl;

                overflow-y: auto;

                overflow-x: hidden;

            }
```

```css
div.content{

    flex: 1;

    width: 100%;

    background-color: white;

    height: 100%;

    color: black;

    display: flex;

    flex-direction: column;

    overflow-x: hidden;

}



li.assignment-button {

    padding: 10px;

    margin: 15px;

    border-radius: 10px;

    text-transform: uppercase;

    text-align: center;

    word-wrap: break-word;

    max-width: 100%;

    border-radius: 20px;

    background-color: #9ccbfe;

    border: 3px solid black;

    cursor: pointer;

}



div.assignment-name {

    font-size: 18px;

    pointer-events: none;

}
```

```css
div.assignment-subject {

    font-size: 14px;

    font-weight: bold;

    pointer-events: none;

}


ul#assignments-list {

    list-style: none;

    margin: 0;

    padding: 0;

}


.sidebar-button {

    background-color: #007BFF;

    border: 0;

    color: #ffffff;

    font-size: 30px;

    font-weight: bold;

    display: flex;

    align-items: flex-start;

    flex-direction: column;

    cursor: pointer;

}


.sidebar-minimized {

    background-color: #007BFF;

    border: 0;

    color: #ffffff;
```

```
        font-size: 30px;

        font-weight: bold;

        display: flex;

        align-items: flex-start;

        flex-direction: column;

    }


    .display-date {

        padding-right: 10px ;

        flex: 1;

        writing-mode: vertical-lr;

        text-align: center;

        font-size: 20px;

        transform: rotate(180deg);

    }


    h1#assignment-name {

        padding-top: 20px;

        font-size: 30px;

        text-align: center;

        text-transform: uppercase;

    }


    #assignment-detail{

        font-size: 20px;

        padding-left: 20px;

        padding-right: 20px;

        text-align: justify;

        white-space: pre-wrap;
```

```
}


#assignment-posted-by{
    font-size: 15px;
    padding-left: 20px;
    padding-right: 20px;
    text-align: justify;
    white-space: pre-wrap;
}


#due{
    color: rgb(166, 36, 36);
    font-size: 28px;
}


#pic{
    vertical-align: top;
    font-size: 18px;
    padding-right: 20px;
}


img{
    width : 60px;
    height : 60px ;
    border-radius: 25%;
}


table#forum{
    text-align: left;
    padding-bottom: 15px;

    overflow-x: hidden;

    box-sizing: border-box;

    margin: 30px;

}


#cmt-box{

    border: 2px solid black;

    border-radius: 20px;

    padding: 15px;

    overflow-y: auto;

    word-wrap: break-word;

    background-color: #b6d9ff;

}


#cmt-box p{

    color: red;

}


#cmt-box , #username{

    font-size: 18px;

}


#time{

    font-size: 15px;

    text-align: right;

    display: block;

}
```

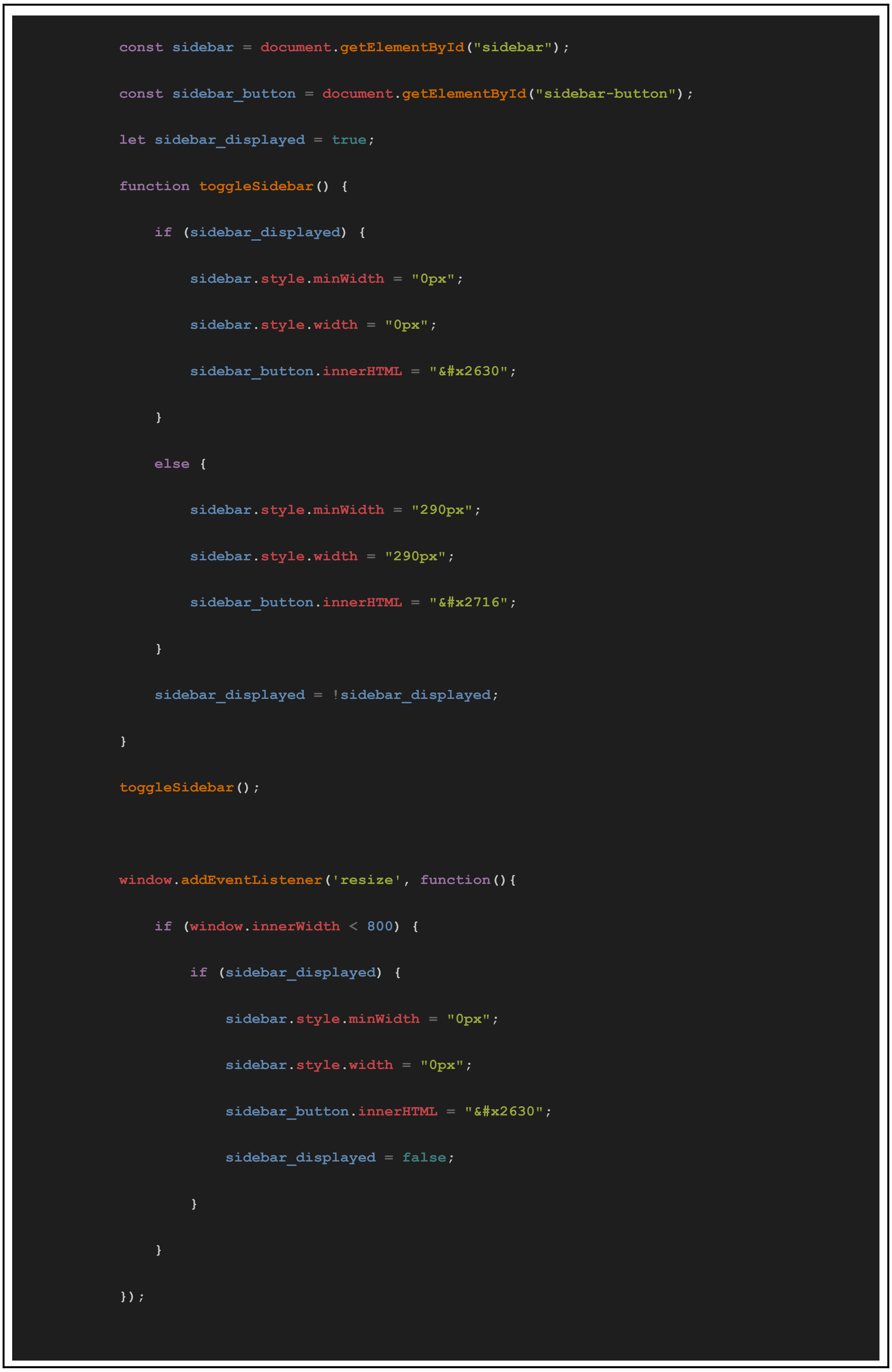
```css
#submit{
    cursor: pointer;

    transition: .2s;

    position: fixed;
    bottom: 25px;
    right : 20px ;
    background-color: #b6d9ff;

    border-radius: 50%;
    border: 2px solid black;
    align-items: center;
    justify-content: center;

    font-size: 30px;

    width : 50px ;
    height: 50px;
    transform: rotate(270deg);
    padding-bottom: 4px;
}

#submit:hover{
    background-color: #007BFF;
    color: white;
}

.space{
    padding-bottom: 100px;

    background-color: white;

}



textarea {

    font-size:  25px;

    font-family: Mali;



    width : 60vw;

    height: 40px;



    border-radius: 10px;



    position: fixed;



    bottom: 30px;

    left: 32vw;



    resize: none;

}



#username{

    cursor: pointer;

}



#reply-indicator {



    display: none;

    position: fixed;
```

```css
    bottom: 70px;
    left: 33vw;

    background-color: #d6d6d6;
    border-radius: 10px  10px 0px 0px;
    padding: 5px;
}

#close-reply-indicator{
    display: none;
    cursor: pointer;

    width: 20px;
    height: 20px;

    background-color: #d6d6d6;
    border: 1px solid black;
    border-radius: 50%;

    position: fixed;

    bottom: 85px;
    left: 32vw;

    font-size: 13px;
    line-height: 15px;
}
```

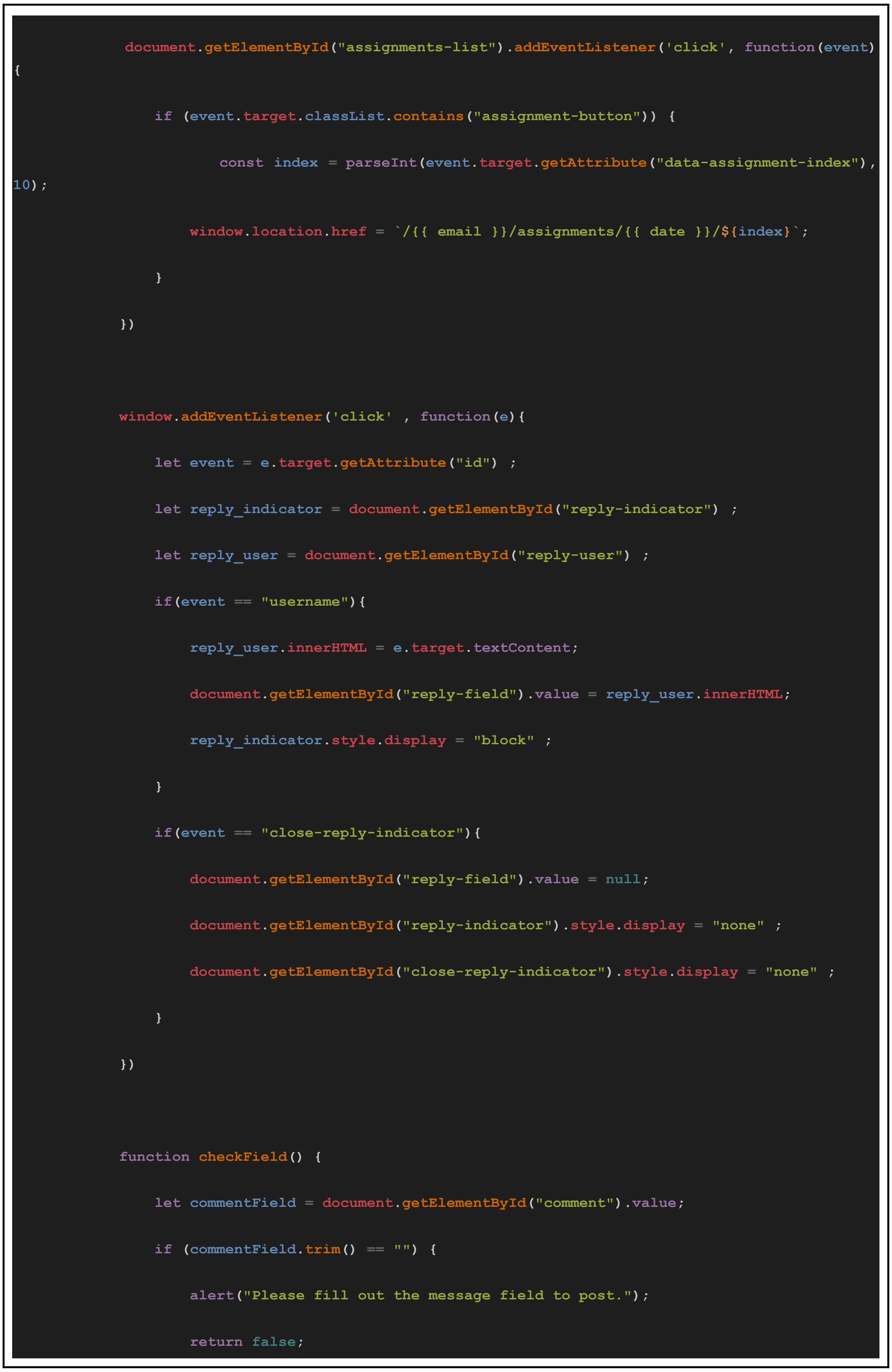
```
        #go-back {
            position: fixed;
            bottom: 0;
            left: 0;
            height: 50px;
            width: 50px;
            border: 0;
            background-color: transparent;
            font-size: 40px;
            color: orange;
            cursor: pointer;
        }

        #go-back:active {
            color: red;
        }
    </style>
</head>
<body onload="on_load()">
    <div class="sidebar" id="sidebar">
        <ul id="assignments-list">
            {% for assignment in assignments %}
                <li class="assignment-button" data-assignment-index="{{ loop.index0 }}">
                    <div class="assignment-subject">{{ assignment.subject }}</div>
                    <div class="assignment-name">{{ assignment.name }}</div>
                </li>
            {% endfor %}
        </ul>
    </div>
```

```
        <div class="sidebar-minimized">
            <button id="sidebar-button" class="sidebar-button" onclick="toggleSidebar()">&#x2716;</button>
            <div class="display-date" id="display-date"></div>
        </div>
        <div class="content">
            <h1 id="assignment-name">{{ assignment.subject }}: {{ assignment.name }}</h1>
            <p id="assignment-detail">{{ assignment.content }}</p>
            <p id="assignment-posted-by">Posted by: {{ assignment.posted_by }}</p>
            <table id="forum">
                {% for message in assignment.forum.values() %}
                    <tr>
                        <td id="username">{{ message.user }}</td>
                    </tr>
                    <tr>
                        <td id="cmt-box">
                            {% if message.reply_user %}
                                <p>RE : {{ message.reply_user }}</p>
                            {% endif %}
                            <div>{{ message.comment }}</div>
                            {% if message.filename %}
                                <a href="/static/{{ message.filename }}" download>Download file</a>
                            {% endif %}
                            <span id="time">{{ message.time }}</span>
                        </td>
                    </tr>
                    <tr style="height: 15px;"></tr>
                {% endfor %}
            </table>
```

```html
            <!-- space -->
            <div class="space"></div>

            <!-- re -->
            <div id="reply-indicator">RE: <span id="reply-user"></span></div>
            <button id="close-reply-indicator">✖</button>

            <form id="post-form" onsubmit="return checkField()" method="post" enctype="multipart/form-data">

                <!-- Text mesage -->
                <input type="hidden" id="reply-field" name="reply_user">
                <textarea id="comment" name="comment"></textarea>

                <!-- File Upload -->
                <input type="file" id="file-upload" name="file">

                <button id = "submit" type="submit">➜</button>
            </form>
        </div>
        <button id="go-back" onclick="goBack()">&#9664;</button>
    </body>
        <script>
            let date = "{{ date }}";
            let formatted_date = new Date(date).toLocaleDateString("en-US", {day: "numeric", month: "long", year: "numeric"});
            const display_date = document.getElementById("display-date");
            display_date.textContent = formatted_date;
```



```
const sidebar = document.getElementById("sidebar");

const sidebar_button = document.getElementById("sidebar-button");

let sidebar_displayed = true;

function toggleSidebar() {

    if (sidebar_displayed) {

        sidebar.style.minWidth = "0px";

        sidebar.style.width = "0px";

        sidebar_button.innerHTML = "&#x2630";

    }

    else {

        sidebar.style.minWidth = "290px";

        sidebar.style.width = "290px";

        sidebar_button.innerHTML = "&#x2716";

    }

    sidebar_displayed = !sidebar_displayed;

}

toggleSidebar();


window.addEventListener('resize', function(){

    if (window.innerWidth < 800) {

        if (sidebar_displayed) {

            sidebar.style.minWidth = "0px";

            sidebar.style.width = "0px";

            sidebar_button.innerHTML = "&#x2630";

            sidebar_displayed = false;

        }

    }

});
```



```
        document.getElementById("assignments-list").addEventListener('click', function(event)
{
            if (event.target.classList.contains("assignment-button")) {
                const index = parseInt(event.target.getAttribute("data-assignment-index"),
10);
                window.location.href = `/{{ email }}/assignments/{{ date }}/${index}`;
            }
        })


        window.addEventListener('click' , function(e){
            let event = e.target.getAttribute("id") ;
            let reply_indicator = document.getElementById("reply-indicator") ;
            let reply_user = document.getElementById("reply-user") ;
            if(event == "username"){
                reply_user.innerHTML = e.target.textContent;
                document.getElementById("reply-field").value = reply_user.innerHTML;
                reply_indicator.style.display = "block" ;
            }
            if(event == "close-reply-indicator"){
                document.getElementById("reply-field").value = null;
                document.getElementById("reply-indicator").style.display = "none" ;
                document.getElementById("close-reply-indicator").style.display = "none" ;
            }
        })


        function checkField() {
            let commentField = document.getElementById("comment").value;
            if (commentField.trim() == "") {
                alert("Please fill out the message field to post.");
                return false;
            }

            if (document.getElementById("file-upload").value != "") {

                let file = document.getElementById("file-upload").files[0];

                if (file.size > 10000000) {

                    alert("File size must not exceed 10MB!");

                    return false;

                }

            }

            return true;

        }



        function goBack() {

            window.location.href = `/{{ email }}/assignments/{{ date }}`;

        }



        function on_load() {

            if ("{{ visual }}" == 'dark_mode')

            {

                document.body.querySelector(".sidebar").style.backgroundColor = "#1a1a1a";

                document.body.querySelector(".sidebar-minimized").style.backgroundColor = "#1a1a1a";

                document.body.querySelector(".sidebar-minimized").style.color = "white";

                document.body.querySelector(".content").style.backgroundColor = "#1a1a1a";

                document.body.querySelector(".content").style.color = "white";

                document.body.querySelector("#sidebar-button").style.color = "white";

                document.body.querySelector("#sidebar-button").style.backgroundColor = "#1a1a1a";

                document.body.querySelector("#go-back").style.color = "white";

                document.body.querySelector("#go-back").style.backgroundColor = "#1a1a1a";

                document.body.querySelector("#submit").style.color = "white";
                document.body.querySelector("#submit").style.backgroundColor = "#1a1a1a";

                document.body.querySelector("#close-reply-indicator").style.color = "white";

                 document.body.querySelector("#close-reply-indicator").style.backgroundColor =
"#1a1a1a";

                document.body.querySelector("#reply-indicator").style.color = "white";

                      document.body.querySelector("#reply-indicator").style.backgroundColor =
"#1a1a1a";

                let td = document.querySelectorAll("td");

                for (let i = 0; i < td.length; i++) {

                    if (td[i].id == "username") {

                        td[i].style.backgroundColor = "#1a1a1a";

                        td[i].style.color = "white";

                    }

                }



                document.body.querySelector(".space").style.backgroundColor = "#1a1a1a";



            }

        }



    </script>



</html>
```

Assignment.html

```
<html>

    <head>

        <meta charset="utf-8">

        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Assignments</title>

        <style>

            body {

                    font-family: Arial, sans-serif;

                    background-color: white;

                    margin: 0;

                    padding: 0;

                    display: flex;

                    flex-direction: row;

                }


            div.header{

                pointer-events: none;

                background-color: #007BFF;

                color: #f4f4f4;



                writing-mode: vertical-lr;

                width : 10vw ;

                text-align: center;

                font-size: calc(min(5vw, 10vh, 50px));



                transform: rotate(180deg);



                border-right: 1px solid #494949;

                padding-right: 2vw;

            }
```

```css
div.content{
    flex: 1;
    background-color: white;
    color: black;
    text-align : center;
    overflow-y: auto;
}

div.topic{
    pointer-events: none;

    font-size: 50px;
    padding-top: 30px;
    padding-bottom: 30px;
    border-bottom: 5px solid rgb(0, 0, 0);
    margin: 50px;
    margin-top: 0px;
}

ul#assignments-list {
    list-style: none;
    margin: 0;
    padding: 0;
    padding-bottom: 5px;
}
```

```css
li.assignment-button {
    padding: 10px;
    text-transform: uppercase;
    text-align: center;
    word-wrap: break-word;
    background-color: #9ccbfe;
    border: 5px solid black;
    min-height: 60px;
    margin: 8vw;
    margin-top: 0px;
    margin-bottom: 10px;
    border-radius: 20px;
    width: 70vw;
    cursor: pointer;
}


.assignment-name {
    font-size: 25px;
    pointer-events: none;
}


.assignment-subject {
    font-size: 22px;
    font-weight: bold;
    pointer-events: none;
}
```

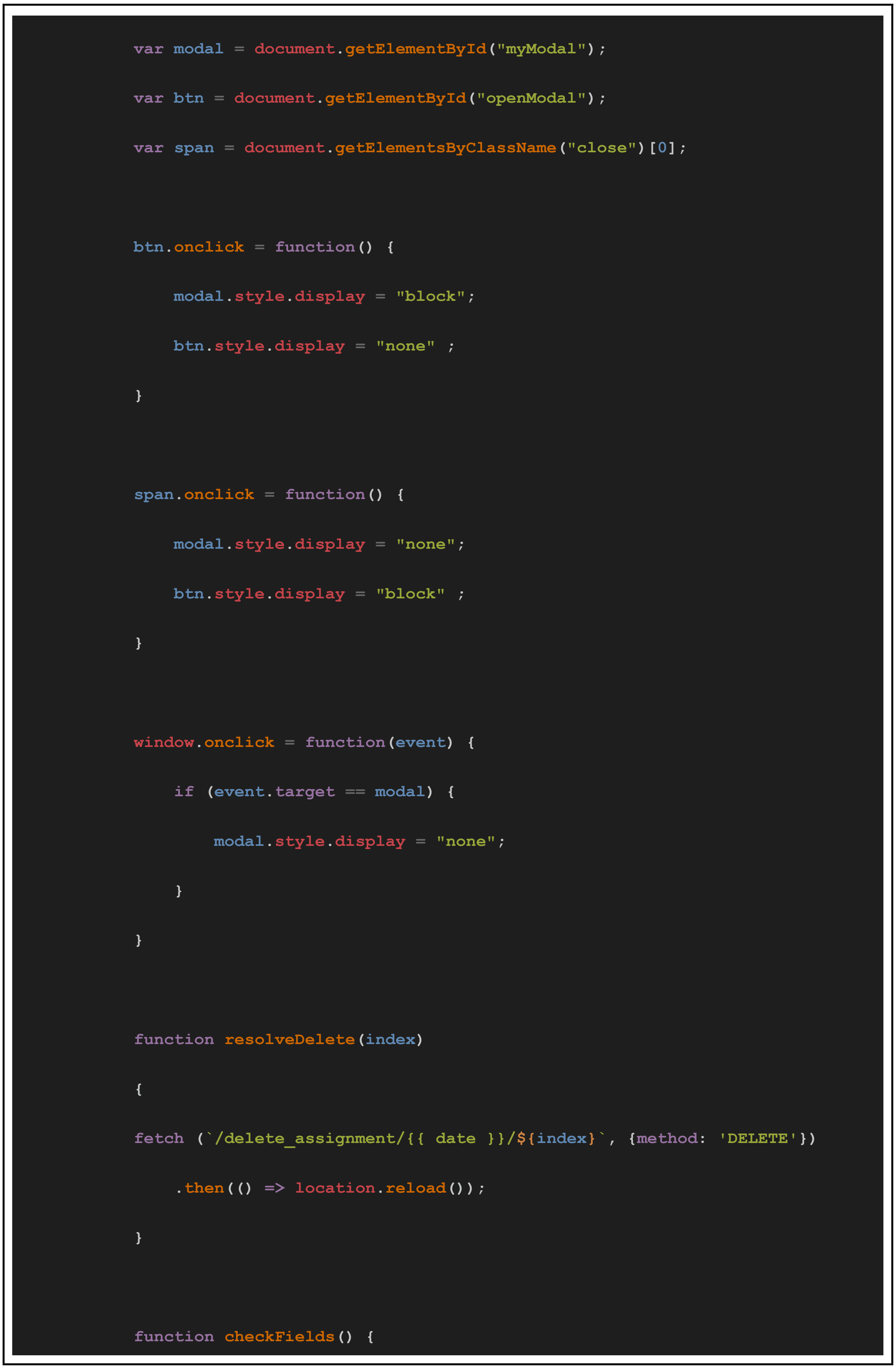
```css
            .modal {
                display: none;
            }

            .close {
                display: flex;
                transition: .2s;
                cursor: pointer;
                font-size: 50px;
                width: fit-content;
                height: fit-content;
            }

            .close:hover{
                color: red;
            }

            #submit , #openModal{
                font-family: Arial, sans-serif;
                font-size: 24px;
                cursor: pointer;
                background-color: #9ccbfe;
                border-radius: 20px;
                display: flex;
                margin-top: 50px;
                left: 50%;
                transition: .2s;
```

```css
    width: 200px;

    margin: 20px auto;

    align-items: center;

    padding: 10px;

}


#submit:hover , #openModal:hover , .close:hover{

    transform: scale(1.1 , 1.1);

}


table{

    width: 100%;

}


td {

    vertical-align: top;

    height: 70px;

    padding-left: 30px;

    color: black;

    font-size: 25px;

}


textarea{

    resize: none;

    font-size: 25px;

    width: 90%;

    border-radius: 10px;
    font-family: Arial, sans-serif;
    height: 150px;
}

select , input {
    font-size:  25px;
    font-family: Arial, sans-serif;
    width : 90% ;
    border-radius: 10px;
}

#go-back {
    position: fixed;
    bottom: 0;
    left: 0;
    height: 50px;
    width: 50px;
    border: 0;
    background-color: transparent;
    font-size: 40px;
    color: orange;
    cursor: pointer;
}

.dark-mode {
    background-color: #333;
    color: #f4f4f4;
```

```
        }

        #go-back:active {
            color: red;
        }

        .delete_asn {
            font-family: Arial, sans-serif;
            font-size: 24px;
            cursor: pointer;
            background-color: #4781c0;
            border-radius: 20px;
        }

        .delete_asn:hover {
            background-color: #333;
        }

    </style>
</head>
<body onload="on_load()">
    <div class="header">
        <p id = "display-date"></p>
    </div>
    <div class="content">
        <div class="topic">Assignments</div>
        <ul id="assignments-list">
```

```
                {% for assignment in assignments %}
                   <li class="assignment-button" data-assignment-index="{{ loop.index0
}}">
                    <div class="assignment-subject">{{ assignment.subject }}</div>
                    <div class="assignment-name">{{ assignment.name }}</div>
                              <button class="delete_asn" name="action" value="Delete"
onclick="resolveDelete({{ loop.index0 }})">Delete</button>
                </li>
            {% endfor %}
            </ul>
            {% if can_edit == True %}
                <button id="openModal">Add Assignment</button>
                <div id="myModal" class="modal">
                    <div class="modal-content">
                        <div class="close">×</div>
                        <form onsubmit="return checkFields()" method="post">
                            <table>
                                <tr>
                                           <td><label for="assignment_name">Assignment
Name:</label></td>
                                           <td><input type="text" id="assignment_name"
name="assignment_name"></td>
                                </tr>
                                <tr>
                                    <td><label for="subject">Subject:</label></td>
                                                  <td><input type="text" id="subject"
name="subject"></td>
                                </tr>
                                <tr>
```

```
                                    <td><label for="content">Details:</label></td>

                                                        <td><textarea id="content"
name="content"></textarea></td>

                                </tr>

                            </table>

                            <input type="submit" value="Add Assignment" id="submit">

                        </form>

                    </div>

                </div>

            {% endif %}

        <button id="go-back" onclick="goBack()">&#9664;</button>

    </body>

        <script>

            let date = "{{ date }}";

                let formatted_date = new Date(date).toLocaleDateString("en-US", {day:
"numeric", month: "long", year: "numeric"});

            const display_date = document.getElementById("display-date");

            display_date.textContent = formatted_date;



                 document.getElementById("assignments-list").addEventListener('click',
function(event) {

                if (event.target.classList.contains("assignment-button")) {

                                                                const index =
parseInt(event.target.getAttribute("data-assignment-index"), 10);

                            window.location.href = `/{{ email }}/assignments/{{ date
}}/${index}`;

                }

            })
```



```
var modal = document.getElementById("myModal");

var btn = document.getElementById("openModal");

var span = document.getElementsByClassName("close")[0];


btn.onclick = function() {

    modal.style.display = "block";

    btn.style.display = "none" ;

}


span.onclick = function() {

    modal.style.display = "none";

    btn.style.display = "block" ;

}


window.onclick = function(event) {

    if (event.target == modal) {

        modal.style.display = "none";

    }

}


function resolveDelete(index)

{

fetch (`/delete_assignment/{{ date }}/${index}`, {method: 'DELETE'})

    .then(() => location.reload());

}


function checkFields() {
```

```
                let nameField = document.getElementById("assignment_name").value;

                let subjectField = document.getElementById("subject").value;

                let contentField = document.getElementById("content").value;

                         if  (nameField.trim()  ==  ""  ||  subjectField.trim()  ==  ""  ||
contentField.trim() == "") {

                    alert("Please fill out all fields to post.");

                    return false;

                }

                return true;

            }



            function goBack() {

                let yyyymm = "{{ date }}".slice(0, -3);

                window.location.href = `/{{ email }}/main/${yyyymm}`;

            }



            function on_load() {

                if ("{{ visual }}" == 'dark_mode') {

                    document.body.classList.add('dark-mode');

                          document.querySelector('div.header').style.backgroundColor  =
'black';

                         document.querySelector('div.header').style.borderRight  =  '1px
solid #f4f4f4';

                         document.querySelector('div.content').style.backgroundColor  =
'#333';

                         document.querySelector('div.topic').style.borderBottom  =  '5px
solid #f4f4f4';

                    document.querySelector('div.topic').style.color = '#f4f4f4';

                           document.querySelector('div.topic').style.backgroundColor  =
'#333';
document.querySelectorAll('li.assignment-button').forEach((assignment_button) => {
                        assignment_button.style.backgroundColor = '#9ccbfe';
                        assignment_button.style.border = '5px solid #f4f4f4';
                    });
                    document.querySelectorAll('td').forEach((td) => {
                        td.style.color = '#f4f4f4';
                    });
                }
            }


        </script>

</html>
```

Error.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Error</title>
</head>
<body>
    <h1>Error</h1>
    <p>{{ error }}</p>

</body>

</html>
```

admin_login

```
<!DOCTYPE html>

<html>

<head>

    <title>Admin Login</title>

</head>

<style>

    body {

        font-family: Arial, sans-serif;

        margin: 0;

        padding: 0;

        background-color: #f0f0f0;

    }


    h1 {

        background-color: #333;

        color: #fff;

        padding: 10px 0;

        text-align: center;
```

```css
}


form {

    display: flex;

    flex-direction: column;

    gap: 10px;

    width: 300px;

    margin: 0 auto;

}


input[type="password"] {

    padding: 2px;

    border-radius: 5px;

    border: 1px solid #ccc;

}


input[type="submit"] {

    padding: 5px 10px;

    border-radius: 5px;

    border: none;

    color: #fff;

    cursor: pointer;

    background-color: #4CAF50;

}
```

```
    .dark-mode {

        background-color: #333;

        color: #fff;

    }


</style>

<body>

    <h1>Admin Login</h1>

    <form action="/admin" method="post">

        <input type="password" id="password" name="password"><br>

        <input type="submit" value="Login">

    </form>

</body>

<script>

    window.addEventListener('load', (event) =>

    {

        if ("{{ visual }}" == 'dark_mode') {

            document.body.classList.add('dark-mode');

        }

    });


</script>

</html>
```

admin_page.html

```
<!DOCTYPE html>

<html>

<head>

    <title>Admin Page</title>

</head>

<style>

    body {

        font-family: Arial, sans-serif;

        margin: 0;

        padding: 0;

        background-color: #f0f0f0;

    }



    h1 {

        background-color: #333;

        color: #fff;

        padding: 10px 0;

        text-align: center;

    }



    ul {
```

```css
    list-style-type: none;
    margin: 0;
    padding: 0;
}

li {
    margin: 10px 0;
}

form {
    display: flex;
    flex-direction: column;
    gap: 10px;
}

input[type="text"] {
    padding: 5px;
    border-radius: 5px;
    border: 1px solid #ccc;
}

input[type="submit"] {
    padding: 5px 10px;
    border-radius: 5px;
    border: none;

    color: #fff;

    cursor: pointer;

    transition: background-color 0.3s ease;

}


input[type="submit"][name="action"][value="Update"] {

    background-color: #4CAF50;

}


input[type="submit"][name="action"][value="Delete"] {

    background-color: #f44336;

}


input[type="submit"][name="action"][value="Create"] {

    background-color: #008CBA;

}


input[type="submit"]:hover {

    background-color: #333;

}


.dark-mode {

    background-color: #333;
```

```
        color: #fff;

    }


</style>

<body>

    <h1>Admin Page</h1>

    <ul>

    {% for key, value in data.items() %}

        <li>

            <h2>{{ key }}</h2>

            <ul>

            {% for item_key, item_value in value.items() %}

                {% if key != "Logs" %}

                <li>

                    <form action="/admin_action" method="post">

                        <label for="key">{{ item_key }}:</label>

                              <!-- <label type="hidden" name="action"
value="Update"> -->

                               <input type="text" id="key" name="key"
value="{{ item_key }}" readonly>

                           <input type="text" id="value" name="value"
value="{{ item_value }}">

                                   <input type="submit" name="action"
value="Update">

                                   <input type="submit" name="action"
value="Delete">
```

```
                    </form>

                </li>

                {% else %}

                <li>{{ item_key }}: {{ item_value }}</li>

                {% endif %}

            {% endfor %}

            {% if key != "Logs" %}

            <li>

                <form action="/admin_action" method="post">

                    <label for="key">Key:</label>

                    <input type="text" id="key" name="key">

                    <label for="value">Value:</label>

                    <input type="text" id="value" name="value">

                                   <input  type="submit"  name="action"
value="Create">

                </form>

            {% endif %}

            </ul>

        </li>

    {% endfor %}

    </ul>



       <form  action="/purge_database"  method="get"  onsubmit="return
confirm('Are you sure you want to purge the database? This cannot be
undone.');">
```

```html
        <input type="submit" value="Purge Database">
    </form>


</body>
<script>
    window.addEventListener('load', (event) => {
        on_load();
    });

    function on_load() {
        console.log("{{ visual }}");
        if ("{{ visual }}" == 'dark_mode') {
            document.body.classList.add('dark-mode');
        }
    }
</script>
</html>
```