**Міністерство освіти і науки України**

**Національний технічний університет України**

**“Київський політехнічний інститут”**

**Факультет прикладної математики**

**Кафедра спеціалізованих комп’ютерних систем**

**Лабораторна робота № 3**

з дисципліни «ВЕБ-дизайн»

«Розробка функціональності Web-додатка мовою Javascript»



Виконав:

студент групи КВ-83

**Глеб Влпдислав**

Київ 2021

**Зміст**Мета: ознайомитись із засобами фреймворка VueJS та навчитись створювати асинхронні запити до Web-сервера

Адаптувати програмний код ЛР№2 до вимог фреймворка VueJS та забезпечити завантаження необхідних даних з Web-сервера

**Завдання**

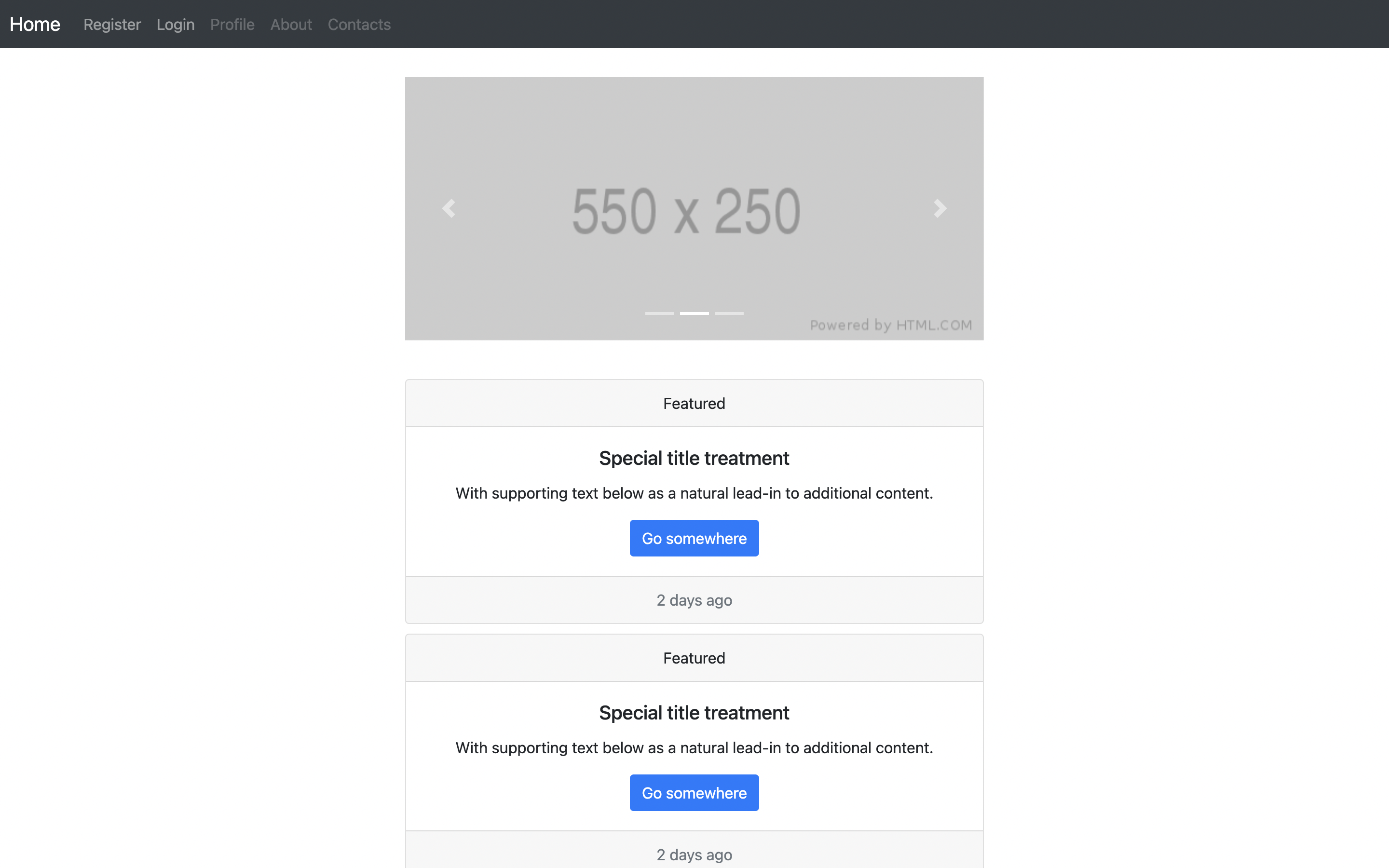
**Мій Telegram:** @vladyslav\_hleb

**Github-репозиторій з лабораторної:** https://github.com/Vlad492/web-lab3

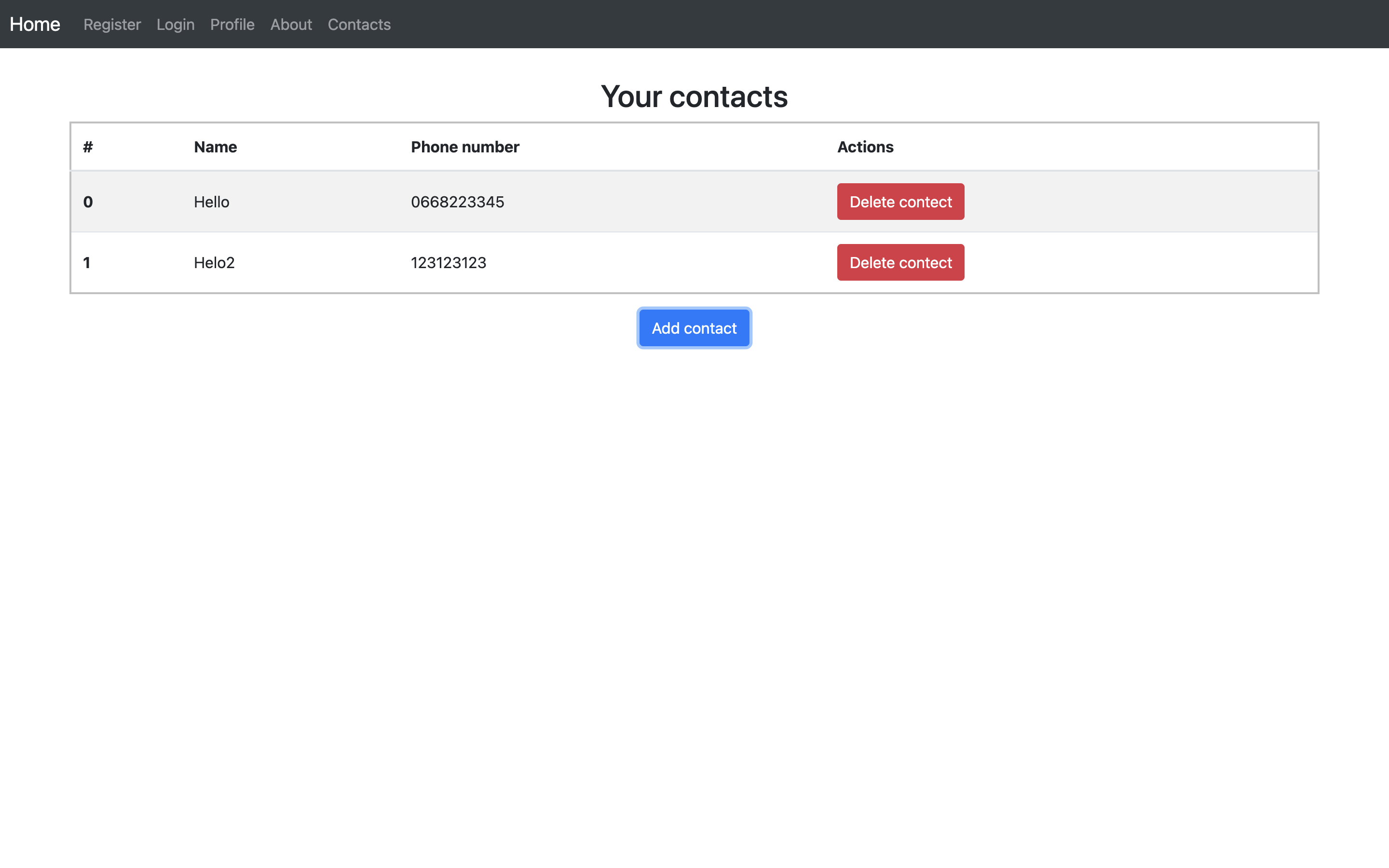
**Тема: “Телефонний довідник.**   
 У цій лабораторній роботі я розробив статичний Web-інтерфейс для додатку “Телефонний довідник”. Були реалізовані наступні веб-сторінки:

* index.html - головна сторінка. На ній користувач може подивитись новини ітд.
* contacts.html - сторінка для перегляду, створення і видалення контактів. На ній знаходиться таблиця з контактами і кнопка для створення контакту.
* profile.html - сторінка профілю користувача.
* about.html - сторінка з детальною інформацією про додаток.
* Реєстрація і вхід винесені в окремі модальні вікна.

**index.html**

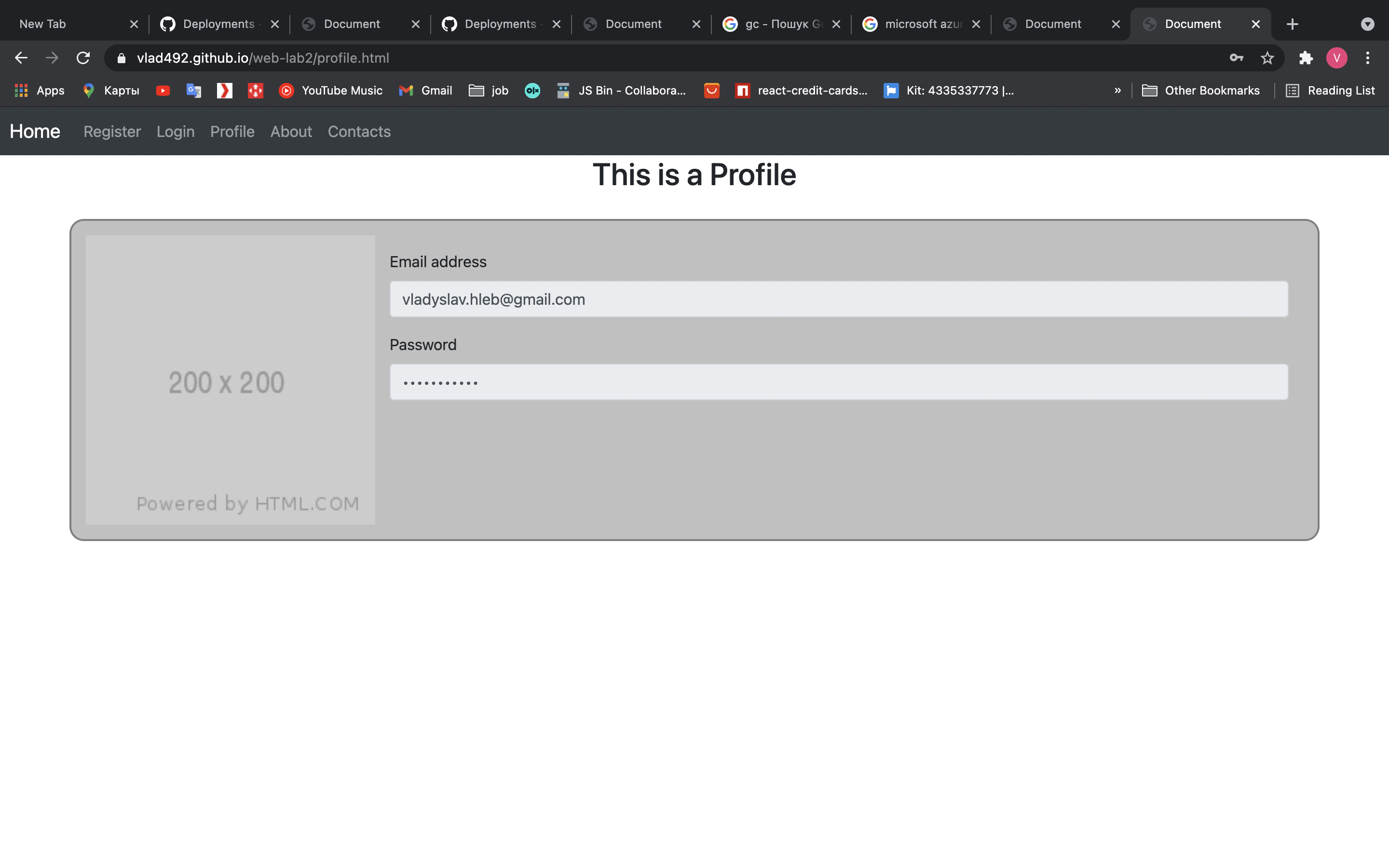


**contacts.html**

****

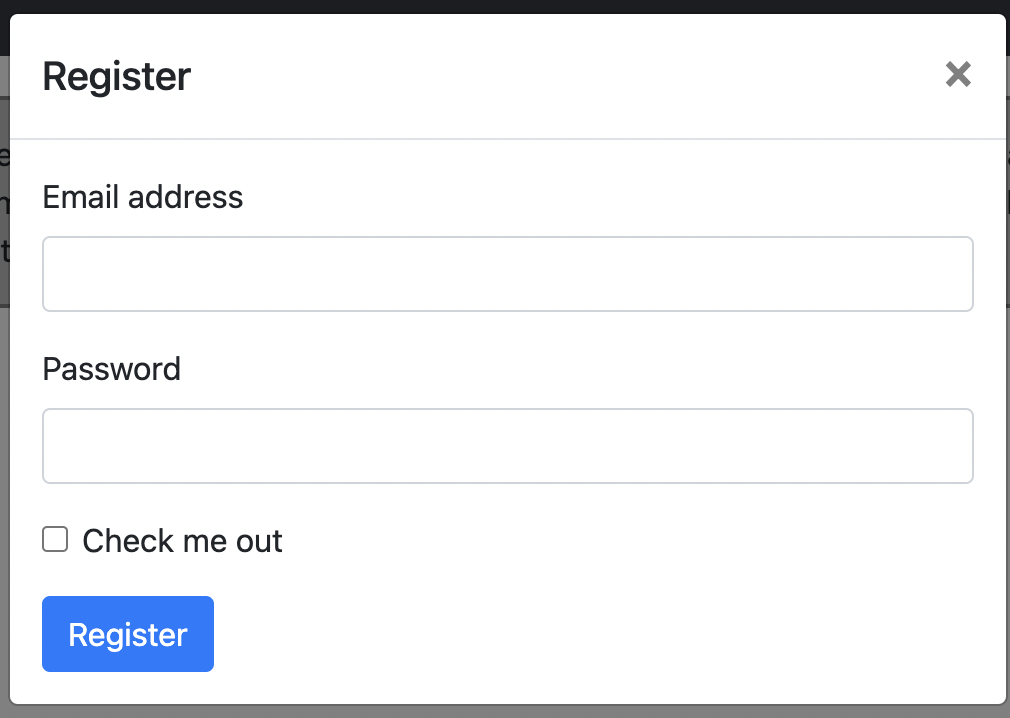
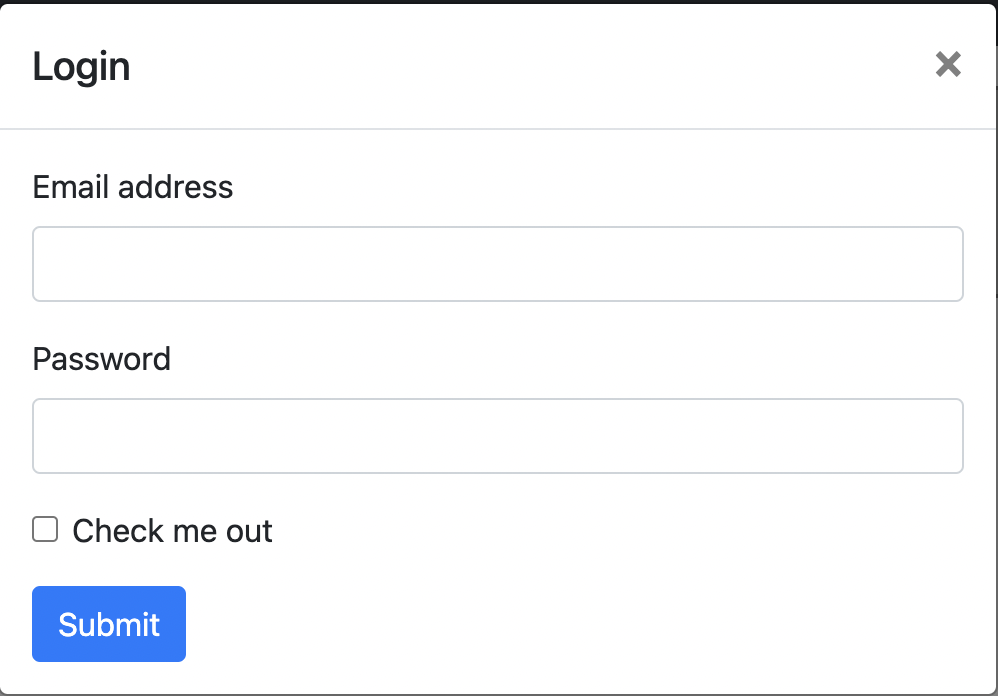
Була реалізована можливість створювати і видаляти контакти. При натисненні на кнопку “Add contact”, контакт буде створено. При натисканні на кнопку “delete contact” контакт буде видалено. Всі контакти зберігаються в localStorage, тому не видаляються після перезавантаження сторінки

**profile.html**

****

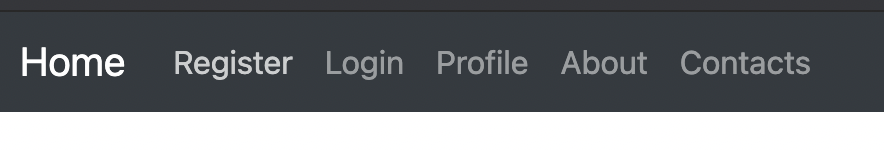
**Login/registration**

У лабораторній роботі було розроблено авторизацію. E кожного юзера є свій email і password, які зберігаються в localStorage.

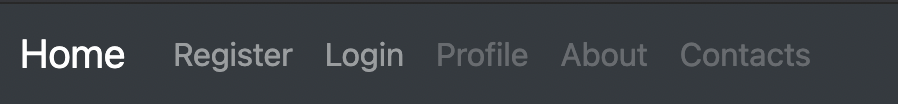
****

Якщо юзер не залогінений то для нього недоступні деякі поля в хедері.

Хедер залогіненого юзера:



Хедер незалогіненого юзера:



**Діаграма класів**

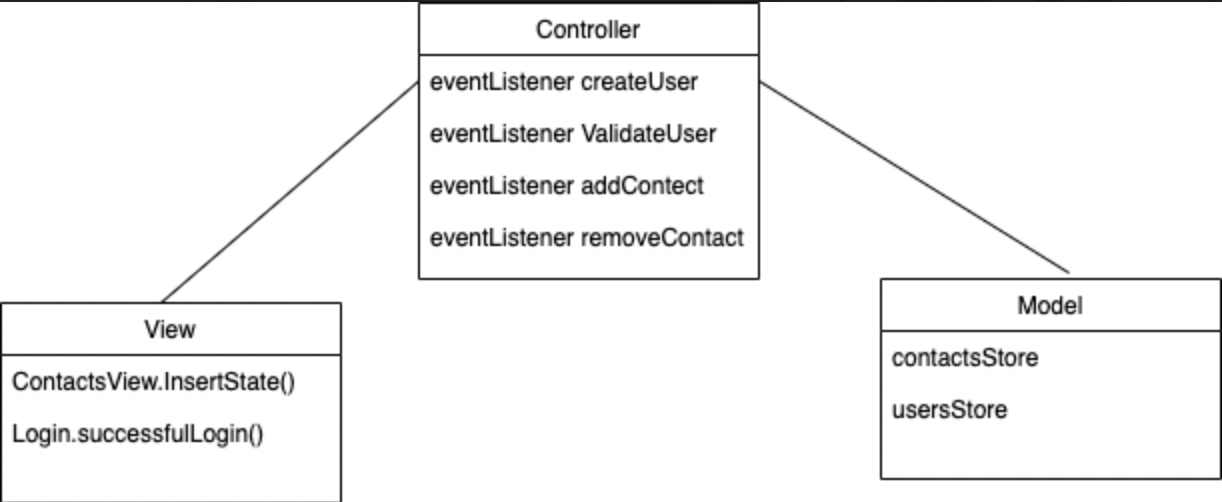
Файл view.js відповідає за таблицю з контактами, при загрузці сайту він підписується на зміни сейта з контактами і виклакається після кожної його зміни(логіка аналогічна бібліотеці Redux).

Файл controller.js навішує прослуховувачі подій на кнопки на сторінках, коли відбувається подія то виконується функція, яка звертається до моделі данних.

Файл users.js ініціалізує стейт користувачів, які можуть залогінитись. В ньому прописаний один action.type, який додає юзера в стейт юзерів

Файл contacts.js ініціалізує стейт контактів. В ньому прописано 2 action.type, які додають і видаляють контакти.

Файл createStore.js – функція яка створює стори, в яких ми можемо зберігати дані,



View.js

class ContactsView {

static insertState() {

var element = document.getElementById("contact-table");

if (element) {

while (element.firstChild) {

element.removeChild(element.firstChild);

}

}

new Vue({

el: '#contact-table',

data: {

contacts: [...contactsStore.getState()]

}

})

}

}

ContactsView.insertState()

class Login {

static successfullLogin() {

let disabledLinks = document.getElementsByClassName('nav-link-login')

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

disabledLinks[i].classList.remove('disabled')

}

}

}

contactsStore.subscribe(() => ContactsView.insertState())

**Сервер**

**Index.js**

const express = require('express')

const app = express()

var bodyParser = require('body-parser')

app.use(bodyParser.json({ type: 'application/\*+json' }))

const { Client } = require('pg')

const client = new Client('postgres://vlad:vlad@localhost:5432/weblab3')

await client.connect()

app.post('/validate',(req,res)=>{

const data = await client.query(`SELECT \* from users where '${req.body.login}'`)

if(data.rows.length > 0 && data.rows[0].password == req.body.password){

res.sendStatus(202)

}else{

res.sendStatus(401)

}

})

app.post('/create',(req,res)=>{

const res = await client.query(`INSERT into users (login,password) values ('${req.body.login},${req.body.password}')`)

res.sendStatus(201)

})

app.listen(process.env.PORT || 3000)