

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

# Кафедра системного програмування та спеціалізованих комп’ютерних систем

**Лабораторна робота № 2**

з дисципліни **Web-дизайн**

Виконав: студент ІІІ курсу

ФПМ групи КВ-12

Петрушин В. Б.

Київ 2024

**Загальне завдання**: розробити функціональність для статичних сторінок Web-додатку першої лабораторної роботи із використанням шаблону MVC

**Інструменти розробки**: мови HTML5, CSS3, Javascript.

**Зауваження**: Web-фреймворки (ReactJS, AngularJS, VueJS) не використовувати!

***Посилання на Github-репозиторій:***

**<https://github.com/Vitya333777/web_lab2>**

***Посилання на Github Pages:***

[**https://vitya333777.github.io/web\_lab2/**](https://vitya333777.github.io/web_lab2/)

***Контакт у Telegram:***

**@vpetrshn**

***Опис додатка Contact’s Book***

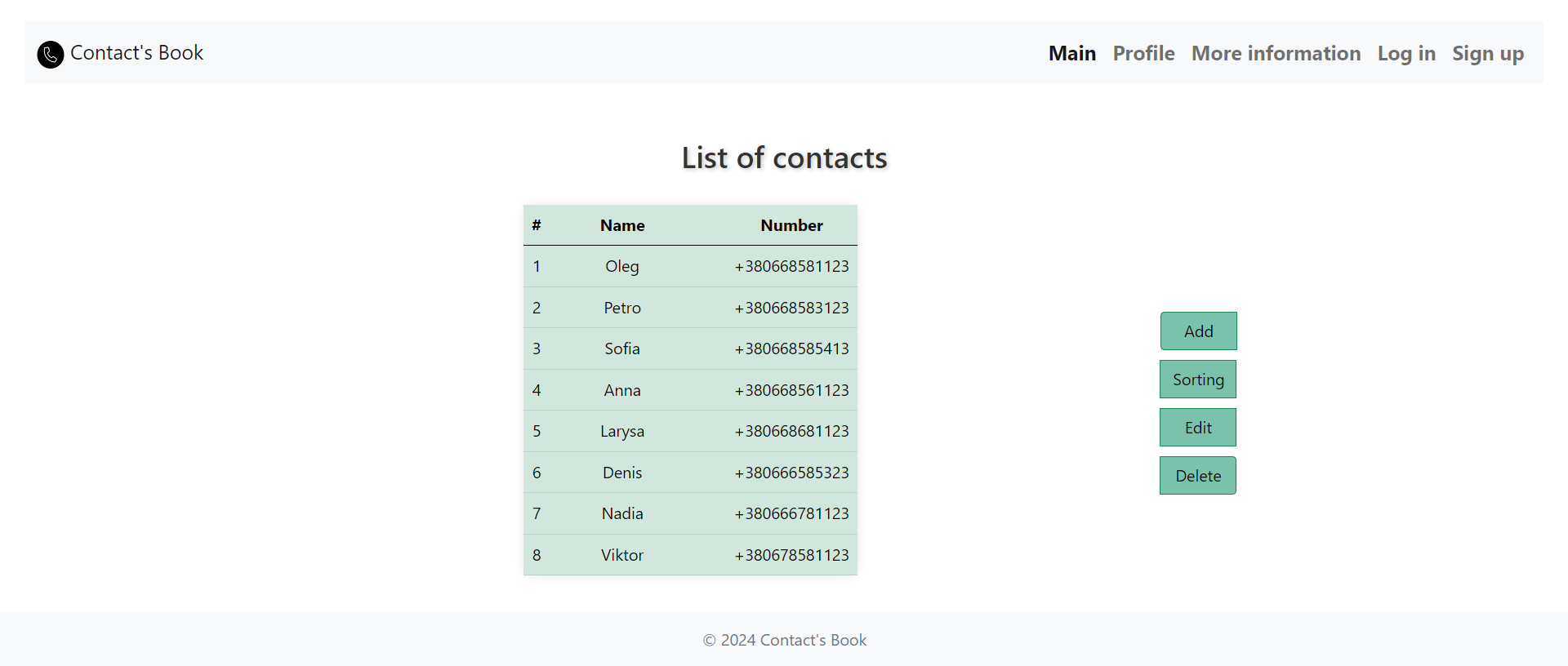
Додаток " Contact’s Book " - це зручний і простий спосіб керувати списком контактів. Його інтерфейс пропонує перегляд, додавання, редагування та видалення контактів у зручному табличному вигляді. Також можна легко відсортувати контакти в алфавітному порядку. Завдяки інтуїтивно зрозумілому користувальницькому інтерфейсу та використанню Bootstrap, додаток забезпечує зручну навігацію та приємний вигляд на будь-якому пристрої.

**Діаграма класів Javascript додатку**

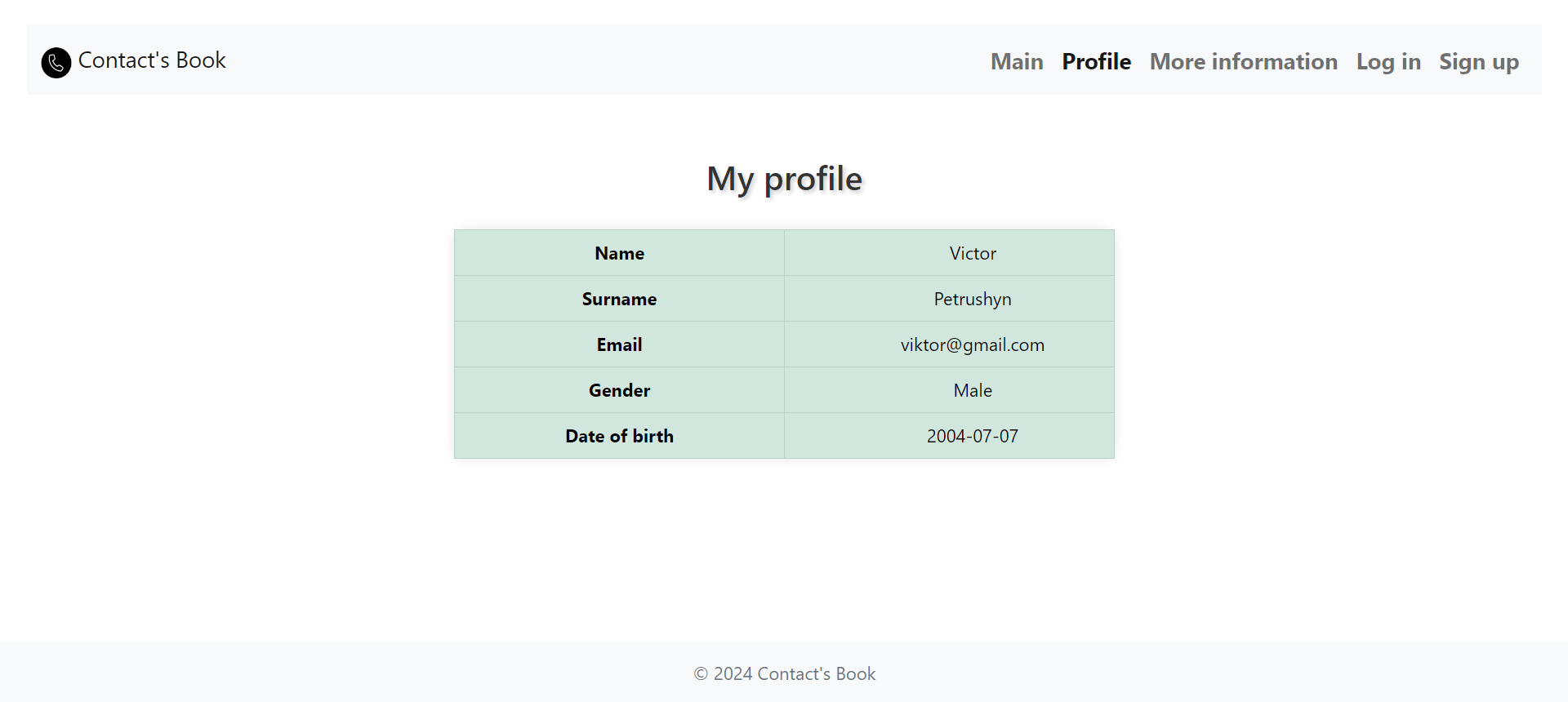
****

***Скріншоти сторінок***

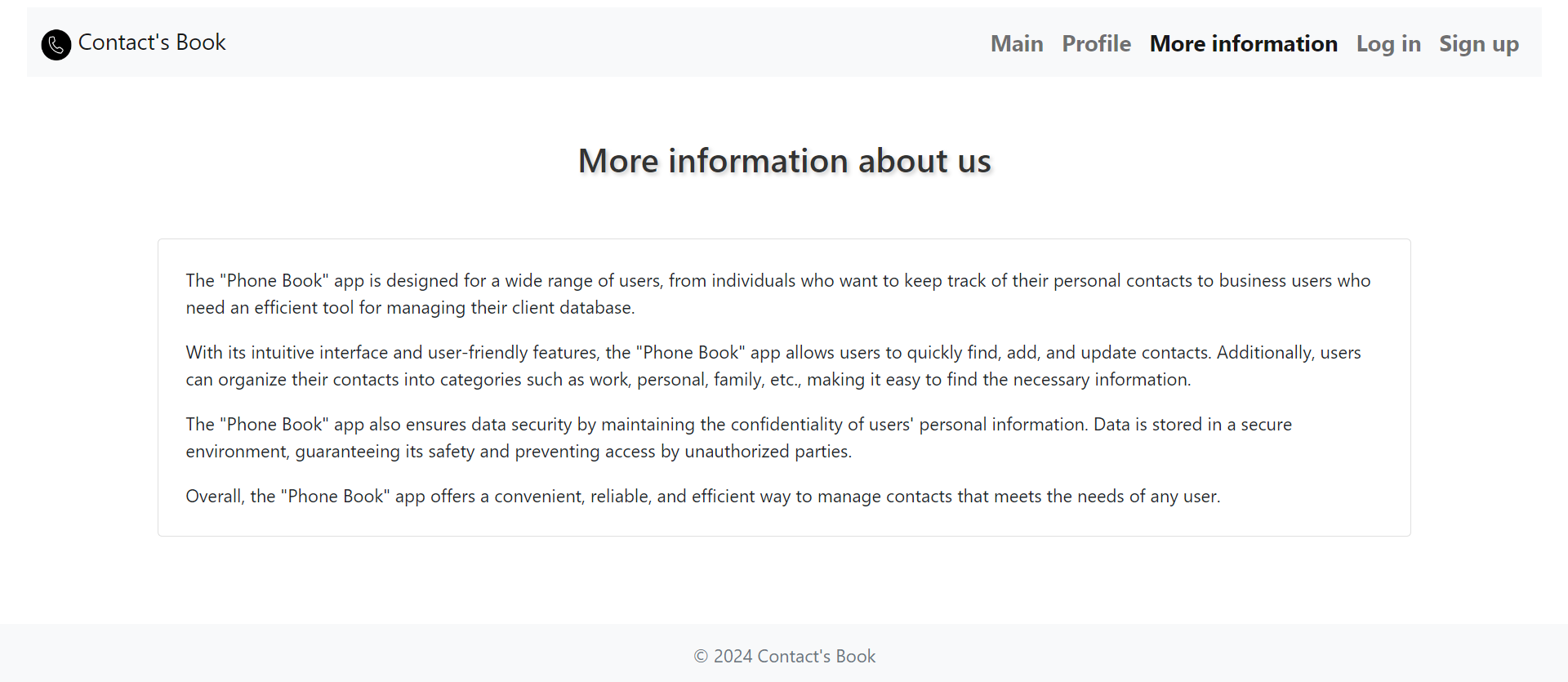
***Main***



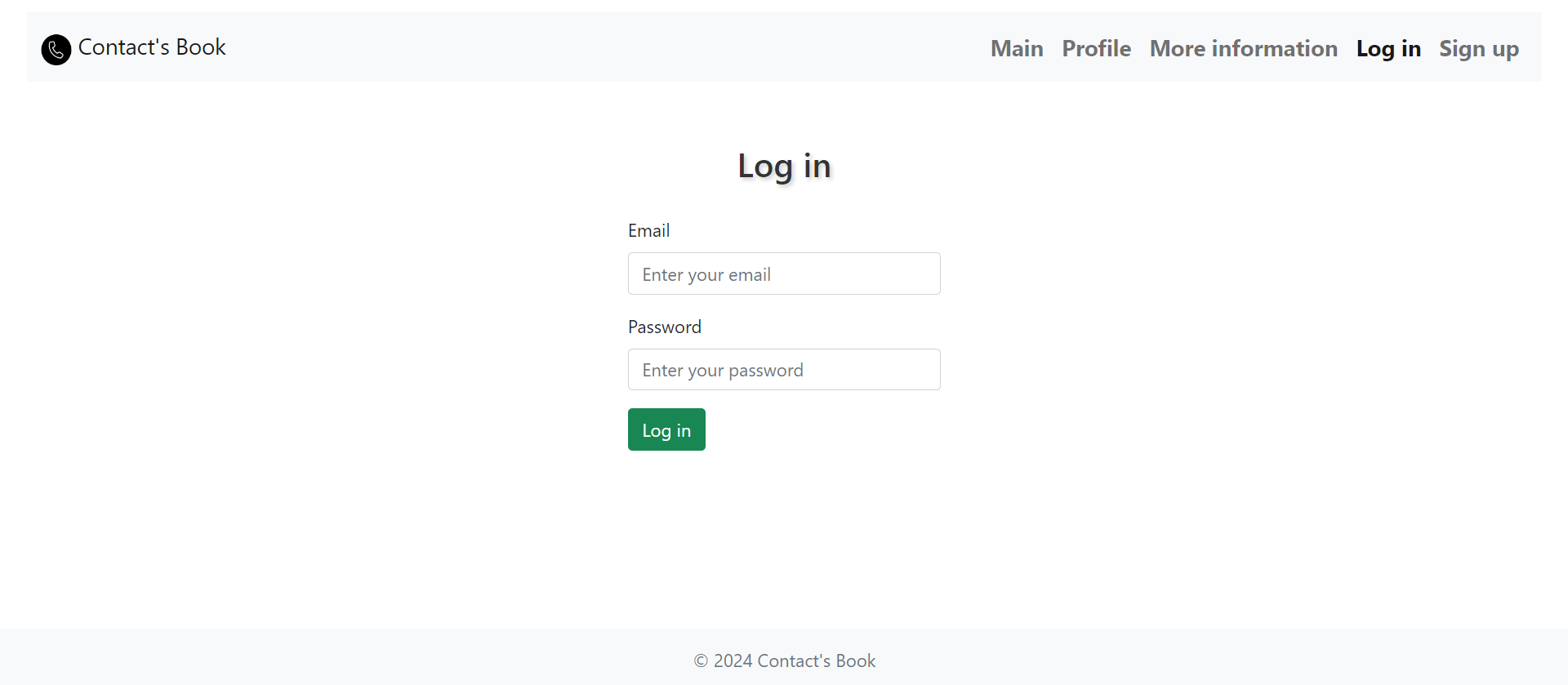
***Profile***



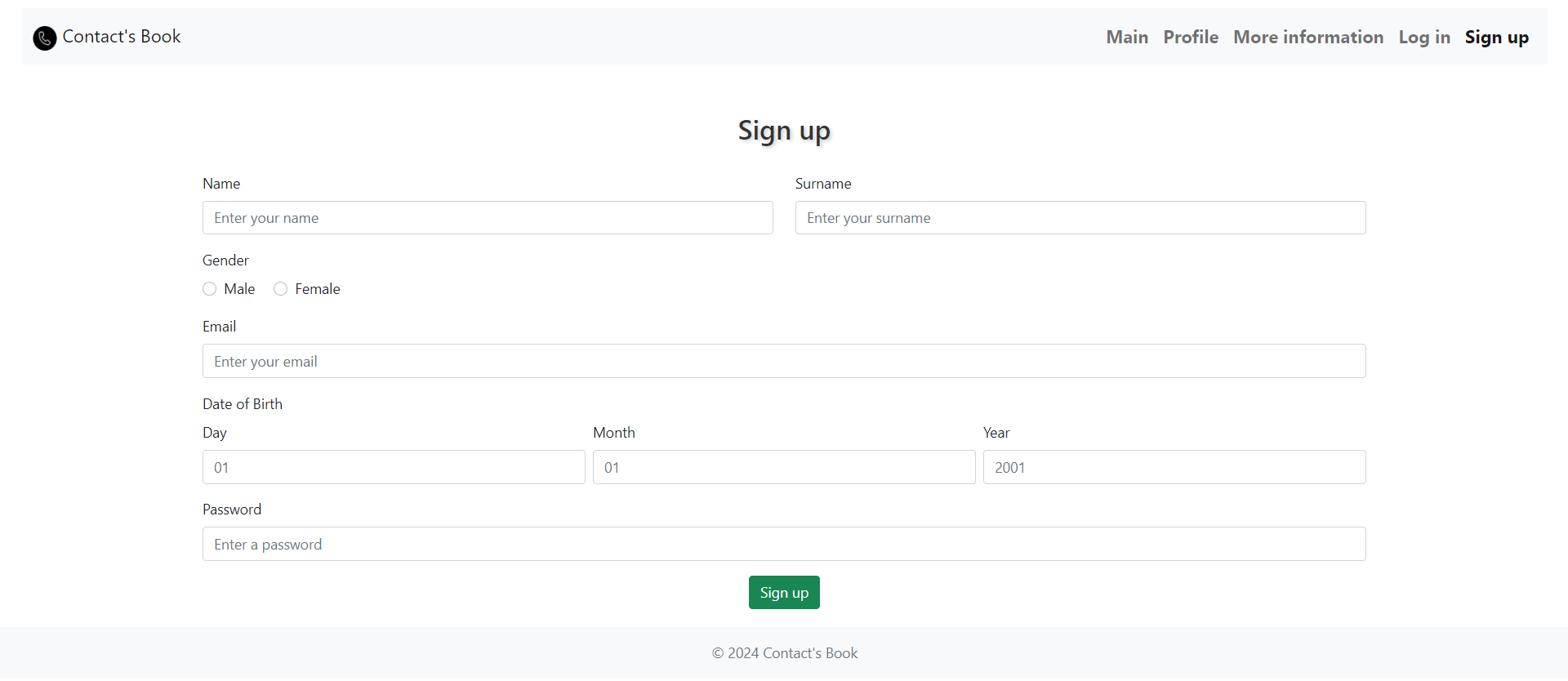
***More information***

******

***Log in***

******

***Sign up***

******

**Вміст Javascript-файлу робочої сторінки**

**model.js**

**const** ContactModel = {

contacts: [],

addContact: **function**(contact) {

**const** tableBody = document.querySelector('tbody');

**const** newRow = ContactView.createTableRow(tableBody.children.**length** + 1, contact.name, contact.number);

tableBody.appendChild(newRow);

$('#addContactModal').modal('hide');

},

getContacts: **function**() {

**return** **this**.contacts;

},

editContact: **function**(id, updatedContact) {

**const** tableRows = document.querySelectorAll('tbody tr');

**const** rowToEdit = tableRows[id - 1];

**if** (rowToEdit) {

rowToEdit.children[1].textContent = updatedContact.name;

rowToEdit.children[2].textContent = updatedContact.number;

$('#editContactModal').modal('hide');

} **else** {

**alert**('Row not found.');

}

},

deleteContact: **function**(id) {

**const** tableBody = document.querySelector('tbody');

**const** rowToDelete = tableBody.querySelector(`tr:nth-child(${id})`);

**if** (rowToDelete) {

rowToDelete.remove();

$('#deleteContactModal').modal('hide');

} **else** {

**alert**('Row not found.');

}

},

sortContacts: **function**(sortBy) {

**const** rowsArray = Array.from(document.querySelector('tbody').children);

rowsArray.**sort**((a, b) => {

**const** textA = a.cells[sortBy === 'name' ? 1 : 2].textContent.**toLowerCase**();

**const** textB = b.cells[sortBy === 'name' ? 1 : 2].textContent.**toLowerCase**();

**return** textA.localeCompare(textB);

});

document.querySelector('tbody').innerHTML = '';

rowsArray.**forEach**(row => document.querySelector('tbody').appendChild(row));

}

};

**const** LoginModel = {

getUserData: **function**() {

**return** localStorage.**getItem**('userData');

}

};

**class** UserData {

constructor(name, surname, gender, email, dateOfBirth, password) {

**this**.name = name;

**this**.surname = surname;

**this**.gender = gender;

**this**.email = email;

**this**.dateOfBirth = dateOfBirth;

**this**.password = password;

**this**.registered = true;

}

saveToLocal() {

localStorage.setItem('userData', JSON.**stringify**(**this**));

}

}

**view.js**

**const** ContactView = {

createTableRow: **function**(index, name, number) {

**const** newRow = document.createElement('tr');

newRow.innerHTML = `

<td>${index}</td>

<td>${name}</td>

<td>${number}</td>

`;

**return** newRow;

},

displayContacts: **function**(contacts) {

**const** tableBody = document.querySelector('tbody');

contacts.**forEach**((contact, index) => {

**const** newRow = ContactView.createTableRow(index + 1, contact.name, contact.number);

tableBody.appendChild(newRow);

});

},

getFormData: **function**(form) {

**const** formData = **new** FormData(form);

**const** name = formData.get('name');

**const** number = formData.get('number');

**return** { name, number };

},

showEditModal: **function**(contact) {

**const** editIdInput = document.getElementById('editId');

**const** editNameInput = document.getElementById('editName');

**const** editNumberInput = document.getElementById('editNumber');

editIdInput.value = contact.id;

editNameInput.value = contact.name;

editNumberInput.value = contact.number;

$('#editContactModal').modal('show');

}

};

document.addEventListener('DOMContentLoaded', () => {

**const** profileTable = document.querySelector('.table-bordered');

**const** storedUserData = localStorage.**getItem**('userData');

**if** (storedUserData) {

**const** userData = JSON.**parse**(storedUserData);

**const** nameCell = profileTable.querySelector('tr:nth-child(1) td:nth-child(2)');

**const** surnameCell = profileTable.querySelector('tr:nth-child(2) td:nth-child(2)');

**const** emailCell = profileTable.querySelector('tr:nth-child(3) td:nth-child(2)');

**const** genderCell = profileTable.querySelector('tr:nth-child(4) td:nth-child(2)');

**const** dateOfBirthCell = profileTable.querySelector('tr:nth-child(5) td:nth-child(2)');

nameCell.textContent = userData.name;

surnameCell.textContent = userData.surname;

emailCell.textContent = userData.email;

genderCell.textContent = userData.gender === 'male' ? 'Male' : 'Female';

dateOfBirthCell.textContent = userData.dateOfBirth;

} **else** {

console.**log**('User data not found. Please log in or sign up first.');

}

});

document.addEventListener('DOMContentLoaded', () => {

**const** registerForm = document.querySelector('.needs-validation');

**if** (registerForm) {

registerForm.addEventListener('submit', (event) => {

event.preventDefault();

event.stopPropagation();

**if** (registerForm.checkValidity()) {

**const** inputName = document.getElementById('inputName').value;

**const** inputSurname = document.getElementById('inputSurname').value;

**const** inputEmail = document.getElementById('registerEmail').value;

**const** inputPassword = document.getElementById('registerPassword').value;

**const** gender = document.querySelector('input[name="gender"]:checked');

**const** inputDay = document.getElementById('inputDay').value;

**const** inputMonth = document.getElementById('inputMonth').value;

**const** inputYear = document.getElementById('inputYear').value;

**if** (!gender) {

**alert**('Виберіть стать');

**return**;

}

**const** dateOfBirth = `${inputYear}-${inputMonth}-${inputDay}`;

**const** userData = {

name: inputName,

surname: inputSurname,

gender: gender.id,

email: inputEmail,

dateOfBirth: dateOfBirth,

password: inputPassword,

registered: true

};

localStorage.setItem('userData', JSON.**stringify**(userData));

**alert**('Успішно зареєстровано✅');

window.location.href = 'index.html';

}

registerForm.classList.add('was-validated');

});

}

**const** loginForm = document.querySelector('form');

**if** (loginForm) {

loginForm.addEventListener('submit', (event) => {

event.preventDefault();

**const** inputEmail = document.getElementById('loginEmail').value;

**const** inputPassword = document.getElementById('loginPassword').value;

**const** storedUserData = localStorage.**getItem**('userData');

**if** (storedUserData) {

**const** userData = JSON.**parse**(storedUserData);

**if** (userData.registered && inputEmail === userData.email && inputPassword === userData.password) {

**alert**('Успішний вхід в акаунт✅');

window.location.href = 'index.html';

} **else** {

**alert**('Неправильний логін чи пароль. Спробуйте ще раз.');

}

} **else** {

**alert**('Користувач не знайдений, зареєструйтесь.');

}

});

}

});

**controller.js**

**const** ContactController = {

init: **function**() {

**const** addContactForm = document.getElementById('addContactForm');

**const** editContactForm = document.getElementById('editContactForm');

**const** applySortingBtn = document.getElementById('applySortingBtn');

**const** deleteBtn = document.getElementById('confirmDeleteBtn');

**const** tableBody = document.querySelector('tbody');

**const** editIdInput = document.getElementById('editId');

**const** editNameInput = document.getElementById('editName');

**const** editNumberInput = document.getElementById('editNumber');

addContactForm.addEventListener('submit', ContactController.AddContact);

editContactForm.addEventListener('submit', ContactController.EditContact);

applySortingBtn.addEventListener('click', ContactController.SortContacts);

deleteBtn.addEventListener('click', ContactController.DeleteContact);

tableBody.addEventListener('click', ContactController.RowClick);

},

AddContact: **function**(event) {

event.preventDefault();

**const** name = document.getElementById('inputName').value;

**const** number = document.getElementById('inputNumber').value;

**if** (name && number) {

**const** newContact = { name, number };

ContactModel.addContact(newContact);

ContactView.displayContacts(ContactModel.getContacts());

$('#addContactModal').modal('hide');

} **else** {

**alert**('Invalid data.');

}

},

EditContact: **function**(event) {

event.preventDefault();

**const** id = **parseInt**(document.getElementById('editId').value);

**const** name = document.getElementById('editName').value;

**const** number = document.getElementById('editNumber').value;

**if** (id && name && number) {

ContactModel.editContact(id, { name, number });

} **else** {

**alert**('Invalid data.');

}

},

SortContacts: **function**() {

**const** criteria = document.querySelector('input[name="sortingCriteria"]:checked');

**if** (criteria) {

**const** sortBy = criteria.value;

ContactModel.sortContacts(sortBy);

} **else** {

**alert**('Please select a sorting criteria.');

}

},

DeleteContact: **function**() {

**const** id = **parseInt**(document.getElementById('deleteId').value);

**if** (id) {

ContactModel.deleteContact(id);

} **else** {

**alert**('Please enter an ID to delete.');

}

},

RowClick: **function**(event) {

**const** rowIndex = event.target.closest('tr').rowIndex - 1;

}

};

ContactController.init();