Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра програмного забезпечення



**ЗВІТ**

**Про виконання лабораторної роботи № 4**

**«Тестування REST Web-service»**

**з дисципліни «Управління якістю ПЗ»**

**Лектор:**

асист. каф. ПЗ

Ваврук І.Є.

**Виконав:**

студ. групи ПЗІП-11

Бойчук Т.В.

**Прийняла:**

асист. каф. ПЗ

Ваврук І.Є.

«\_\_\_» \_\_\_\_\_\_ 2020 р.

∑ = \_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_

Львів – 2020

**ТЕМА РОБОТИ**: Тестування REST Web-service.

**МЕТА РОБОТИ**: Навчитися тестувати REST Web-service.

**ЗАВДАННЯ**

1. Написати REST сервіс відповідно до варіанту
2. Написати REST-клієнт відповідно до варіанту.
3. Протестувати всі методи, використовуючи дата провайдер.
4. Здійснити легування усіх тестів в текстовий документ.
5. Сформувати звіт.

**Варіант 2**

## MailService

1. Створити веб-сервіс «MailService», який емулює роботу mail-сервера з можливістю: 1)відсилати (добавляти одного листа в список листів на сервері), 2)приймати (віддавати список всіх листів на сервері), 3)видаляти лист на сервері, 4) отримувати листи з сервера по мейлу відправника, 5) отримувати листи з сервера по заголовку.

В листі має бути subj, email, body. Обробники помилок на випадок відсутності листа/-ів з відповідними повідомленнями.

1. Створити клієнт веб-сервісу «MailService». Створити тести для мінімум 10 кейсів з обробкою очікуваного результату. Всі операції мають логуватися в лог-file та результат тестування має бути виведений в репорт.

### РЕЗУЛЬТАТИ ВИКОНАННЯ

Для виконання лабораторної роботи створено REST сервіс MailService. Код реалізації сервісу наведено нижче:

public class MailService {  
  
 private ArrayList<Email> emailList;  
  
 public MailService() {  
 this.emailList = new ArrayList<Email>();  
 }  
  
 public String reset() {  
 return "OK";  
 }  
  
 public Email add(Email newEmail) {  
 emailList.add(newEmail);  
 return newEmail;  
 }  
  
 public ArrayList<Email> findByEmail(Email Obj) {  
 ArrayList<Email> secondList = new ArrayList<Email>();  
  
 for( Email a : emailList) {  
 if (a.getEmail().equals(Obj.getEmail())) {  
 secondList.add(a);  
 }  
 }  
 return secondList;  
 }  
  
 public ArrayList<Email> findBySubj(Email Obj) {  
 ArrayList<Email> secondList = new ArrayList<Email>();  
  
 for( Email a : emailList) {  
 if (a.getSubj().equals(Obj.getSubj())) {  
 secondList.add(a);  
 }  
 }  
 return secondList;  
 }  
  
 public String findById(Email Obj) {  
  
 for( Email a : emailList) {  
 if (a.getId().equals(Obj.getId())) {  
 return "FOUND";  
 }  
 }  
 return "NOT FOUND";  
 }  
  
 public String deleteById(Email Obj) {  
 int i =0;  
 for( Email a : emailList) {  
 if (a.getId().equals(Obj.getId())) {  
 break;  
 }  
 i++;  
 }  
 if(i<emailList.size()) {  
 emailList.remove(i);  
 return "DELETED";  
 }  
 else{  
 return "Email is not exist";  
 }  
 }  
  
 public ArrayList<Email> getAll() {  
 return emailList;  
 }  
  
 public String deleteAll() {  
 emailList.clear();  
 return "OK";  
 }  
  
 public String test() {  
 return java.util.UUID.*randomUUID*().toString();  
 }  
}

Також був розроблений клієнтський застосунок за допомогою бібліотеки React

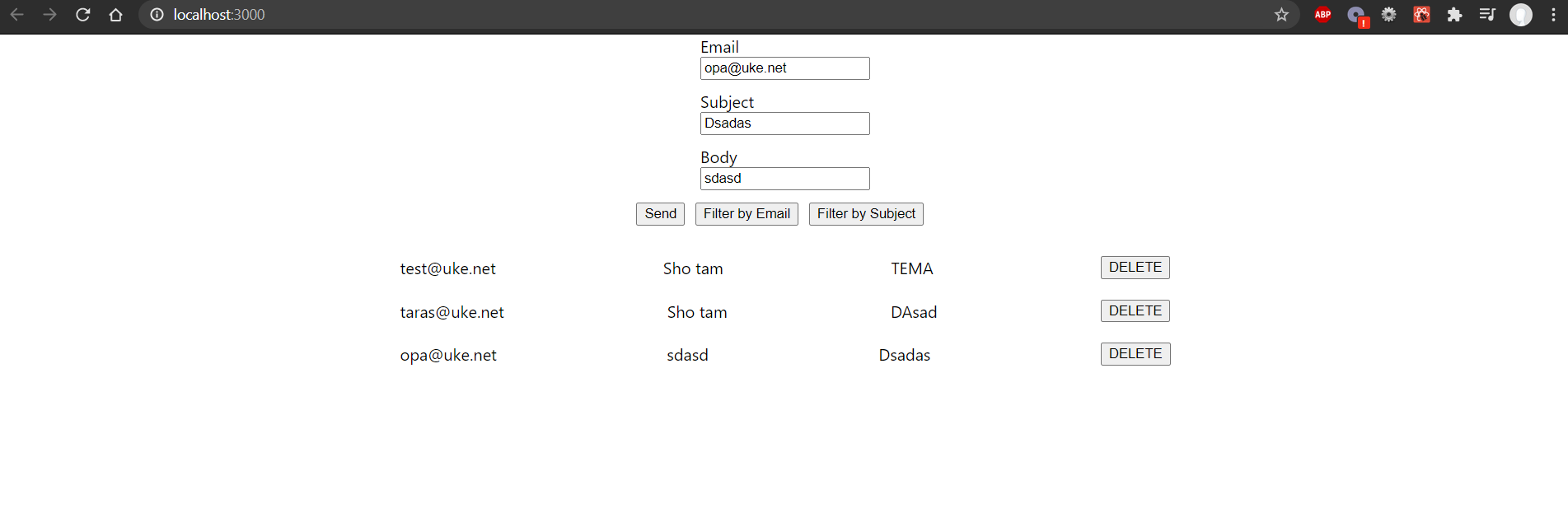
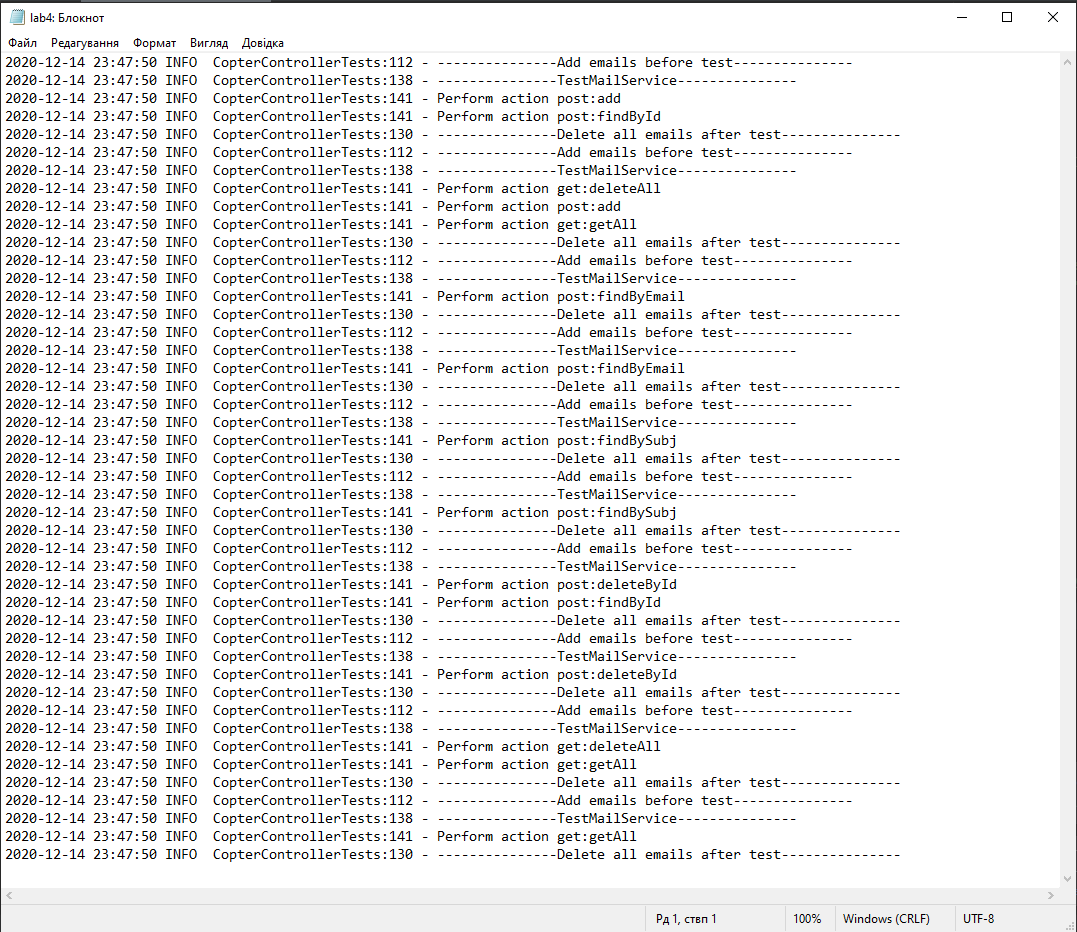


Рис. 1. Скріншот роботи REST-клієнту

**Тестові випадки:**

@WebAppConfiguration  
@SpringBootTest  
@AutoConfigureMockMvc  
public class CopterControllerTests extends AbstractTestNGSpringContextTests {  
 private Logger logger = Logger.*getLogger*(CopterControllerTests.class);  
 @Autowired  
 private MockMvc mockMvc;  
  
 @DataProvider(name = "dataFor\_testMailService")  
 public Object[][] getData() {  
 return new Object[][]{  
 {  
 new ArrayList<String>(Arrays.*asList*("post:add","post:findById")),  
 new ArrayList<String>(Arrays.*asList*(*asJsonString*(new Email("test@ykr.net", "TestBody", "TestSubj", "11")),"FOUND")),  
 new ArrayList<Email>(Arrays.*asList*(new Email("test@ykr.net", "TestBody", "TestSubj", "11"),new Email("", "", "", "11"))),  
 },  
 {  
 new ArrayList<String>(Arrays.*asList*("get:deleteAll","post:add","get:getAll")),  
 new ArrayList<String>(Arrays.*asList*("OK",*asJsonString*(new Email("test@ykr.net", "TestBody", "TestSubj", "11")),*asJsonString*(Arrays.*asList*(new Email("test@ykr.net", "TestBody", "TestSubj", "11"))))),  
 new ArrayList<Email>(Arrays.*asList*(new Email(), new Email("test@ykr.net", "TestBody", "TestSubj", "11"),new Email())),  
 },  
 {  
 new ArrayList<String>(Arrays.*asList*("post:findByEmail")),  
 new ArrayList<String>(Arrays.*asList*(*asJsonString*(Arrays.*asList*(new Email("email1@mail.com", "Sho ti", "Hurma", "001"), new Email("email1@mail.com", "I vprincipi nichoho", "Zal", "002"))))),  
 new ArrayList<Email>(Arrays.*asList*(new Email("email1@mail.com", "", "TestSubj", ""))),  
 },  
 {  
 new ArrayList<String>(Arrays.*asList*("post:findByEmail")),  
 new ArrayList<String>(Arrays.*asList*("[]")),  
 new ArrayList<Email>(Arrays.*asList*(new Email("unexpected@mail.com", "", "TestSubj", ""))),  
 },  
 {  
 new ArrayList<String>(Arrays.*asList*("post:findBySubj")),  
 new ArrayList<String>(Arrays.*asList*(*asJsonString*(Arrays.*asList*(new Email("email1@mail.com", "I vprincipi nichoho", "Zal", "002"), new Email("email3@mail.com", "Hello boys", "Zal", "003"))))),  
 new ArrayList<Email>(Arrays.*asList*(new Email("", "", "Zal", ""))),  
 },  
 {  
 new ArrayList<String>(Arrays.*asList*("post:findBySubj")),  
 new ArrayList<String>(Arrays.*asList*("[]")),  
 new ArrayList<Email>(Arrays.*asList*(new Email("", "", "UnexpectedSubj", ""))),  
 },  
 {  
 new ArrayList<String>(Arrays.*asList*("post:deleteById", "post:findById")),  
 new ArrayList<String>(Arrays.*asList*("DELETED", "NOT FOUND")),  
 new ArrayList<Email>(Arrays.*asList*(new Email("", "", "", "002"), new Email("", "", "", "002"))),  
 },  
 {  
 new ArrayList<String>(Arrays.*asList*("post:deleteById")),  
 new ArrayList<String>(Arrays.*asList*("Email is not exist")),  
 new ArrayList<Email>(Arrays.*asList*(new Email("", "", "", "111"))),  
 },  
 {  
 new ArrayList<String>(Arrays.*asList*("get:deleteAll", "get:getAll")),  
 new ArrayList<String>(Arrays.*asList*("OK", "[]")),  
 new ArrayList<Email>(Arrays.*asList*(new Email(), new Email())),  
 },  
 {  
 new ArrayList<String>(Arrays.*asList*("get:getAll")),  
 new ArrayList<String>(Arrays.*asList*(*asJsonString*(Arrays.*asList*(new Email("email1@mail.com", "Sho ti", "Hurma", "001"),  
 new Email("email1@mail.com", "I vprincipi nichoho", "Zal", "002"),  
 new Email("email3@mail.com", "Hello boys", "Zal", "003"))))),  
 new ArrayList<Email>(Arrays.*asList*(new Email())),  
 },  
 };  
 }  
  
 @BeforeMethod  
 public void doBeforeTest() throws Exception {  
 logger.info("---------------Add emails before test---------------");  
 ArrayList<Email> emailArray = new ArrayList<Email>(Arrays.*asList*(  
 new Email("email1@mail.com", "Sho ti", "Hurma", "001"),  
 new Email("email1@mail.com", "I vprincipi nichoho", "Zal", "002"),  
 new Email("email3@mail.com", "Hello boys", "Zal", "003")  
 ));  
  
 for(int i=0; i<emailArray.size();i++) {  
 this.mockMvc.perform(*post*("/add")  
 .content(*asJsonString*(emailArray.get(i)))  
 .contentType(MediaType.*APPLICATION\_JSON*))  
 .andDo(*print*())  
 .andExpect(*status*().isOk());  
 }  
 }  
  
 @AfterMethod  
 public void doAfterTest() throws Exception {  
 logger.info("---------------Delete all emails after test---------------");  
  
 this.mockMvc.perform(*get*("/deleteAll"))  
 .andExpect(*status*().isOk());  
 }  
  
 @Test(dataProvider = "dataFor\_testMailService")  
 public void testDataProvidedCopter(ArrayList<String> requests,ArrayList<String> responds,ArrayList<Email> contents) throws Exception{  
 logger.info("---------------TestMailService---------------");  
  
 for(int i=0; i<requests.size();i++){  
 logger.info("Perform action " + requests.get(i));  
 if (requests.get(i).split(":")[0].equals("post")){  
 this.mockMvc.perform(*post*("/"+requests.get(i).split(":")[1])  
 .content(*asJsonString*(contents.get(i)))  
 .contentType(MediaType.*APPLICATION\_JSON*))  
 .andDo(*print*())  
 .andExpect(*status*().isOk())  
 .andExpect(*content*().string(responds.get(i)));  
 }  
 else if (requests.get(i).split(":")[0].equals("get")){  
 this.mockMvc.perform(*get*("/"+requests.get(i).split(":")[1]))  
 .andDo(*print*())  
 .andExpect(*status*().isOk())  
 .andExpect(*content*().string(responds.get(i)));  
 }  
 }  
 }  
  
 public static String asJsonString(final Object obj) {  
 try {  
 return new ObjectMapper().writeValueAsString(obj);  
 } catch (Exception e) {  
 throw new RuntimeException(e);  
 }  
 }  
}

Рис. 1. Лог-файл тестування

### ВИСНОВКИ

На даній лабораторній роботі було реалізовано то протестовано REST сервіс згідно варіанту. Під час виконання було написано сам сервіс та його клієнт. До сервісу також було реалізовано тестові випадки. Результати тестування було залоговано за занесено в звіт.