НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Кафедра автоматики та управлiння в технiчних системах

**Курсова робота**

3 дисципліни «Компоненти програмної інженерії - 3. Якість програмного

забезпечення та тестування»

Тема: Розробка інтелектуальної системи розпізнавання людини по голосу

Керівник Виконавець

ас. Хмелюк М.С. ст. Яцук О.Д.

зал. книжка № ІТ-7129

«Допущений до захисту» гр. ІТ-71

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(особистий підпис керівника) (особистий підпис виконавця)

«\_\_» \_\_\_\_\_\_\_\_\_\_\_\_2020р. «\_\_» \_\_\_\_\_\_\_\_\_\_\_\_2020р.

Захищений з оцінкою

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оцінка)

Члени комісії:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(особистий підпис) (розшифровка підпису)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(особистий підпис) (розшифровка підпису)

Київ – 2020

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«Київський політехнічний інститут імені Ігоря Сікорського»

Кафедра АВТОМАТИКИ ТА УПРАВЛІННЯ В ТЕХНІЧНІХ СИСТЕМАХ

Дисципліна «Основи програмування»

Курс 4 Група ІТ-71 Семестр VII

**ЗАВДАННЯ**

**на курсову роботу студента**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Яцука Олександра Дмитровича \_\_\_

1.Тема роботи Розробка інтелектуальної системи розпізнавання людини по голосу \_\_\_

2. Строк здачі студентом закінченої роботи 15.12.2020 \_\_\_

3. Вихідні дані до роботи:

мова програмування С#, .net framework, база даних MSSql, веб-сервер IIS, середній час відгуку не повинен\_перевищувати 0.2 сек, gitlab у якості CI tool, Angular\_\_\_\_\_

4. Зміст розрахунково – пояснювальної записки (перелік питань, що підлягають розробці)

розробити основний веб-інтерфейс користувача з використанням Angular framework, розробити основну серверну частину з використанням .net framework, розробити мікросервіс для розпізнавання людини по голосу, розробити мікросервіс аутентифікації \_\_\_

5.Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

Use-Case, діаграми компонент, станів, послідовностей для 2 мікросервісів та data-flow діаграма \_\_\_

6. Дата видачі завдання 07.10.2020 \_\_\_

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Назва етапів виконання курсової роботи | Термін виконання етапів роботи | Підписи або примітки |
| 1. | Видача завдання | 7.10.2020 |  |
| 2. | Отримання та узгодження теми курсової роботи | 20.10.2020 |  |
| 3. | Архітектура проекту | 25.10.2020 |  |
| 4. | BDD сценарії | 30.10.2020 |  |
| 5. | Розбивка по спринтам (agile) | 1.11.2020 |  |
| 6. | Розробка необхідних діаграм | 5.11.2020 |  |
| 7. | Написання BDD тестів | 8.11.2020 |  |
| 8. | Юніт-тестування мікросервісів | 15.11.2020 |  |
| 9. | Кодування програми | 22.11.2020 |  |
| 10. | Написання інтеграційних тестів | 24.11.2020 |  |
| 11. | Стрес тестування системи | 30.11.2020 |  |
| 12. | Автотести (CI/CD) | 2.12.2020 |  |
| 13. | Розробка діалового інтерфейсу програми | 2.12.2020 |  |
| 14. | Selenium – тестування веб-застосунку | 6.12.2020 |  |
| 15. | Налагодження та перевірка програми | 9.12.2020 |  |
| 16. | Оформлення пояснювальної записки | 13.12.2020 |  |
| 17. | Захист курсової роботи | 15.12.2020 |  |

Студент Яцук О.Д.

(підпис) (прізвище, ім’я, по батькові)

Керівник Хмелюк М.С.

(підпис) (прізвище, ім’я, по батькові)

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2020р

АНОТАЦІЯ

Яцук О.Д. Розробка додатку для розпізнавання людей по голосу. КПІ ім. Ігоря

Сікорського, Київ, 2020.

Робота містить 35 с. тексту, 7 рисунків, 3 таблиці, посилання на 22

літературні джерела, додатки.

Об’єктом розробки є додаток для розпізнавання людини по голосу.

Основною метою програми є створення сервісу який дасть змогу додати ще один рівень безпеки, що буде базуватись на такому біометричному параметрі як голос, до будь-якого застосунку. Окрім цього, не менш важливим пунктом у роботі над курсовою було створення проекту, під час розробки якого можна було ознайомитись з нюансами все можливого тестування програмного застосунку.

У курсовій роботі розроблено бібліотеку для вирішення вище поставленої задачі, а також інтерфейс для взаємодії з нею. Взаємодіяти зі створеними функціями можна як і через публічний Web API, так і використовуючи, розроблену за допомогою Angular Framework, front-end частину.

Основний функціонал програми написаний на мові програмування C#, яка є частиною .NET Framework та з використанням бази даних MS SQL Server. Щодо використання створеного сервісу, то він може бути корисним у тих моментах коли необхідно додати ще один рівень захисту у застосунку. Наявність публічного Web API дає гнучкість представленому рішенню, що дозволяє необмежено використовувати розроблений функціонал у різного типу застосунках, наприклад, мобільні додатки, програми для ПК (персональний комп’ютер) і веб застосунки.

ЗМІСТ

[ВСТУП 6](#_Toc58710071)

[1 ПОСТАНОВКА ЗАДАЧІ 6](#_Toc58710072)

[2 ПРОЕКТУВАННЯ СИСТЕМИ 6](#_Toc58710073)

[2.1 Use-Case діаграма 6](#_Toc58710074)

[2.2 Діаграма компонент 6](#_Toc58710075)

[2.3 Data-flow 6](#_Toc58710076)

[2.4 Діаграма станів 7](#_Toc58710077)

[2.5 Діаграма послідовності 8](#_Toc58710078)

[2.5.1 Діаграма послідовності для сервісу авторизації 8](#_Toc58710079)

[2.5.2 Діаграма послідовності для сервісу ідентифікації по голосу 9](#_Toc58710080)

[2.6 BDD сценарії 10](#_Toc58710081)

[3 АРХІТЕКТУРА ПРОЕКТУ 11](#_Toc58710082)

[3.1 Архітектура сервісу аутентифікації 11](#_Toc58710083)

[3.2 Архітектура сервісу ідентифікації по голосу 11](#_Toc58710084)

[4 ОРГАНІЗАЦІЯ РОБОТИ ПО СПРІНТАХ 11](#_Toc58710085)

[5 ТЕСТУВАННЯ 11](#_Toc58710086)

[5.1 BDD тести 11](#_Toc58710087)

[5.2 Юніт-тести 11](#_Toc58710088)

[5.3 Інтеграційні тести 12](#_Toc58710089)

[5.4 Стрес-тестування 12](#_Toc58710090)

[5.5 Автотести при коміті 12](#_Toc58710091)

[5.6 Selenium тестування 12](#_Toc58710092)

[ВИСНОВКИ 12](#_Toc58710093)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 12](#_Toc58710094)

[ДОТАТОК А 12](#_Toc58710095)

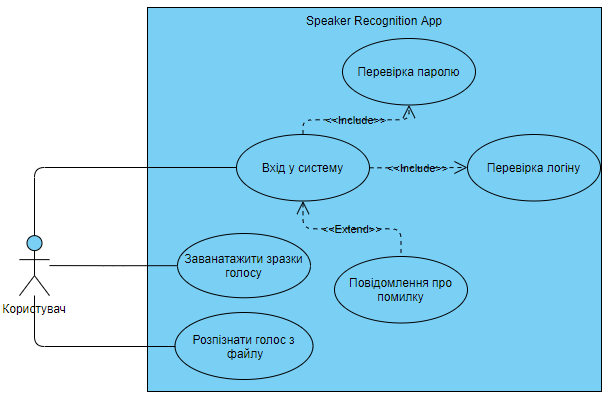
[ДОДАТОК Б 12](#_Toc58710096)

# ВСТУП

# 1 ПОСТАНОВКА ЗАДАЧІ

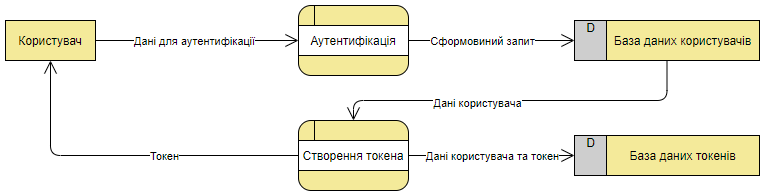
# 2 ПРОЕКТУВАННЯ СИСТЕМИ

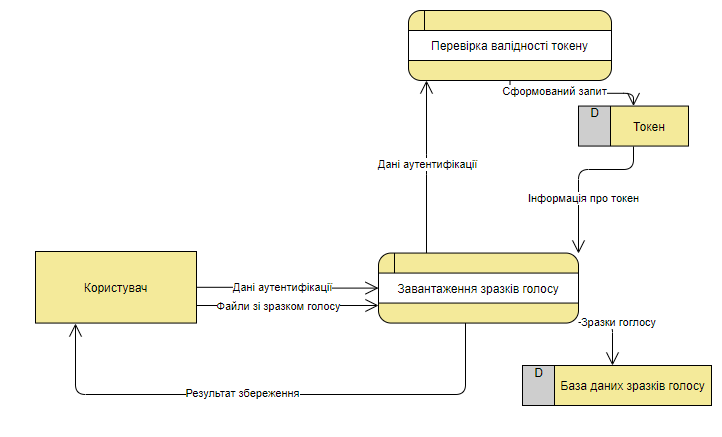
## 2.1 Use-Case діаграма

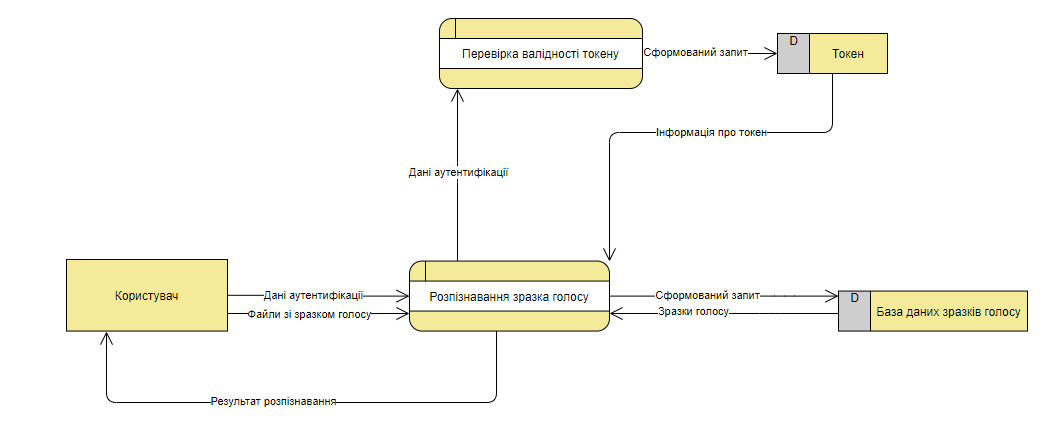


## 2.2 Діаграма компонент

## 2.3 Data-flow



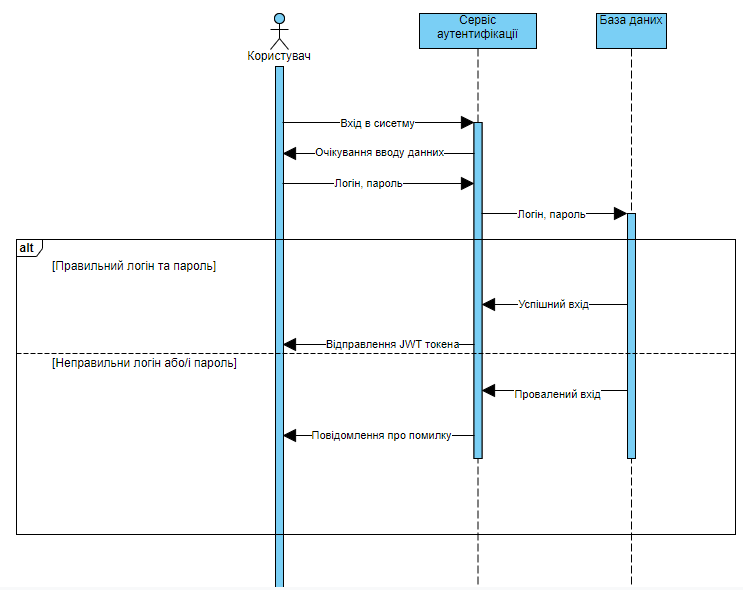




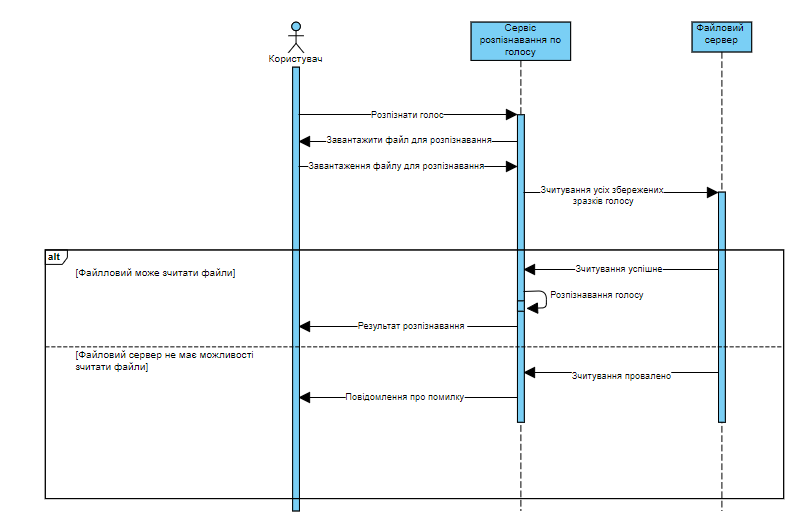
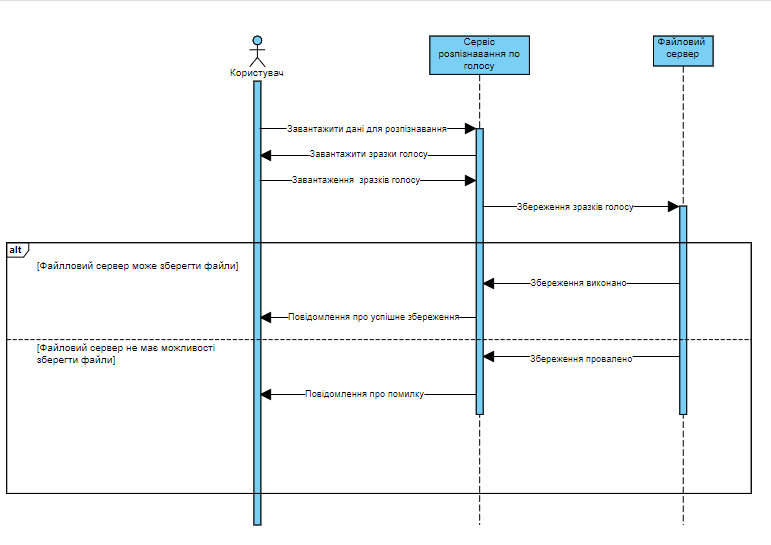
## 2.4 Діаграма станів

## 2.5 Діаграма послідовності

## 2.5.1 Діаграма послідовності для сервісу авторизації



## 2.5.2 Діаграма послідовності для сервісу ідентифікації по голосу



## 2.6 BDD сценарії

Сценарій: Користувач виконує аутентифікацію з правильними даними

Дано користувач знаходиться на сторінці аутентифікації

Коли користувач водить правильний логін і пароль

І натискає кнопку «Log In»

Тоді користувач переходить на основну сторінку додатку

Сценарій: Користувач виконує аутентифікацію з неправильними даними

Дано користувач знаходиться на сторінці аутентифікації

Коли користувач водить неправильний логін і/або пароль

І натискає кнопку «Log In»

Тоді система відображає повідомлення про помилку

Сценарій: Користувач завантажує зразки голосу у систему

Дано користувач знаходиться на сторінці завантаження зразків голосу

Коли користувач натискає кнопку «Завантажити»

Тоді користувач переходить у діалог вибору файлів

Коли користувач вибирає необхідні файли

І натискає кнопку «Ok»

Тоді обрані файли завантажуються у систему

Сценарій: Користувач розпізнає голос з файлу

Дано користувач знаходиться на сторінці розпізнавання голосу

Коли користувач натискає кнопку «Розпізнати»

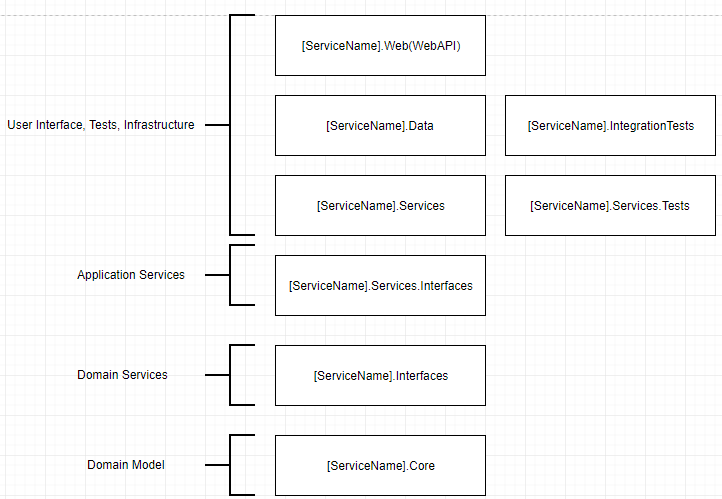
Тоді користувач переходить у діалог вибору файлу

Коли користувач обирає необхідний файл для розпізнавання

І натискає кнопку «Оk»

Тоді система відображає у повідомленні результат розпізнавання голосу

# 3 АРХІТЕКТУРА ПРОЕКТУ

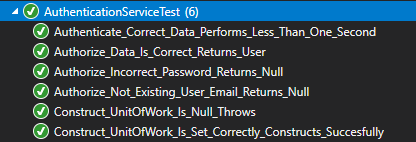


# 4 ОРГАНІЗАЦІЯ РОБОТИ ПО СПРІНТАХ

# 5 ТЕСТУВАННЯ

## 5.1 BDD тести

## 5.2 Юніт-тести



MSTest

[TestMethod]

public void Construct\_UnitOfWork\_Is\_Null\_Throws()

{

//arrange

IUnitOfWork uow = null;

//act

Action act = () => { var sut = new AuthenticationService(uow); };

//assert

act.Should()

.Throw<ArgumentNullException>();

}

## 5.3 Інтеграційні тести

iisexpress

## 5.4 Стрес-тестування

## 5.5 Автотести при коміті

## 5.6 Selenium тестування

6 ІНСТРУКЦІЯ ПРОГРАМІСТА

7 КЕРІВНИЦТВО КОРИСТУВАЧА

# ВИСНОВКИ

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

# ДОТАТОК А

# ДОДАТОК Б

Код програми

Сервіс аутентифікації

using System.ComponentModel.DataAnnotations;

namespace Authentication.Core

{

public class BaseObject

{

[Key]

public string Id { get; set; }

}

}

namespace Authentication.Core

{

public class TokenIdentity : BaseObject

{

public string Token { get; set; }

public string UserId { get; set; }

}

}

namespace Authentication.Core

{

public class User : BaseObject

{

public string Name { get; set; }

public string Email { get; set; }

public string Password { get; set; }

}

}

using System;

using System.Linq;

using Authentication.Core;

using Authentication.Interfaces;

using Authentication.Services.Intefaces;

namespace Authentication.Services

{

public class AuthenticationService : IAuthenticationService

{

private readonly IUnitOfWork \_uow;

public AuthenticationService(IUnitOfWork uow)

{

if (uow == null)

throw new ArgumentNullException(nameof(uow));

\_uow = uow;

}

public User Autheticate(string email, string password)

{

if (string.IsNullOrEmpty(email))

throw new ArgumentNullException(nameof(email));

if (string.IsNullOrEmpty(password))

throw new ArgumentNullException(nameof(password));

var authentificatedUser = \_uow.Users.GetAll()

.FirstOrDefault(x => x.Email == email && x.Password == password);

if (authentificatedUser != null)

{

var ids = \_uow.TokenIdentities.GetAll().Where(x => x.UserId == authentificatedUser.Id).Select(x => x.Id).ToList();

ids.ForEach(x => \_uow.TokenIdentities.Delete(x));

}

return authentificatedUser;

}

public void SaveTokenIdentity(TokenIdentity tokenIdentity)

{

\_uow.TokenIdentities.Create(tokenIdentity);

\_uow.TokenIdentities.Save();

}

public bool Validate(string token)

{

return \_uow.TokenIdentities.GetAll().Where(x => x.Token == token).Any();

}

}

}

Сервіс розпізнавання по голосу

Фрагменти коду для додавання нового зразку голосу у систему

public VoicePrint CreateVoicePrint(T userKey, Stream voiceSampleFile)

{

var audioSample = ConvertFileToDoubleArray(voiceSampleFile);

return CreateVoicePrint(userKey, audioSample);

}

public VoicePrint CreateVoicePrint(T userKey, double[] voiceSample)

{

lock (\_lock)

{

if (userKey == null)

{

throw new ArgumentNullException(nameof(userKey), "The userKey is null");

}

if (store.ContainsKey(userKey))

{

throw new ArgumentException("The userKey already exists: [{userKey}");

}

double[] features = audioProcessor.ProcessAndExtract(voiceSample);

VoicePrint voicePrint = new VoicePrint(features);

if (!universalModelWasSetByUser)

{

if (universalModel == null)

{

universalModel = new VoicePrint(voicePrint);

}

else

{

universalModel.Merge(features);

}

}

store.Add(userKey, voicePrint);

return voicePrint;

}

}