Capstone Design Project 2

**Dance learning application utilizing pose estimation**

Department of Computer Science at the college of IT

School of Computer Science and Engineering

Bartosz Han

Marcel Kućmierowski

Grzegorz Siudak

Wiktoria Stachera

Supervised by professor Seungho Kim and Kyong Kim

October, 2023

**Kyungpook National University**

Table of contents

[Project description 3](#_Toc151239593)

[Defining requirements and specifications 3](#_Toc151239594)

[Features of a good software 3](#_Toc151239595)

[Development 3](#_Toc151239596)

[Methodologies and technologies 4](#_Toc151239597)

[Strategy 4](#_Toc151239598)

[The architecture 5](#_Toc151239599)

[Use cases 5](#_Toc151239600)

[Sequence diagram 5](#_Toc151239601)

[Functional requirements 5](#_Toc151239602)

[Non-functional requirements 5](#_Toc151239603)

[Natural language specification 5](#_Toc151239604)

[Story 5](#_Toc151239605)

[Scenario 5](#_Toc151239606)

[System context 5](#_Toc151239607)

[Designing 5](#_Toc151239608)

[Back-end 5](#_Toc151239609)

[Scoring system 6](#_Toc151239610)

[Implementation 6](#_Toc151239611)

[Front-end 6](#_Toc151239612)

[Testing 6](#_Toc151239613)

[Choreographies 7](#_Toc151239614)

[Verification 7](#_Toc151239615)

[Validation 7](#_Toc151239616)

[Maintenance 7](#_Toc151239617)

[Overall output 7](#_Toc151239618)

[Challenges 7](#_Toc151239619)

[Unresolved issues 7](#_Toc151239620)

[Future improvements 7](#_Toc151239621)

[What is next? 7](#_Toc151239622)

# Project description

Today, many people are interested in dance and decided to practice it with tutorials that can be found on the Internet. However, these tutorials do not provide the necessary feedback to continue training properly. The idea of the project is to create an application whose primary purpose is to provide a review of the performance of the user-dancer compared to a given recorded video, using the recording of the user’s dance and image processing. This application aims to allow dancers to get computed-generated feedback, a missing part of video tutorials, and an excellent opportunity to improve their skills quickly. With our application, dancers can become self-learners while still having objective performance feedback. Dancers can achieve more with our application than with only video tutorials.

# Defining requirements and specifications

### Features of a good software

Today's projects aim to meet the software life cycle process fully. Based on Software Engineering by Ian Sommerville, we revised our work and checked for current practices that helped us improve our approach. Our program provides overall correctness of output and compliance with user requirements. It is easy to maintain and propose changes. It provides reliability, security and safety, efficiency (including resource efficiency), ease of use, and ergonomics.

### Development

First, it is good to establish which development method to choose. We were thinking about waterfall and iterative (*Figure 1*). We decided that iterative (and incremental) design better suited our perspective. Each person could work almost independently, the drawbacks would not cost much time and changes, and we could show partial results to the client (professor at cyclical meetings). As the main idea behind this process states, developing a system took place through repeated cycles and small increments improved by earlier parts of program versions. We could not foresee all the capabilities or features needed, so our program was evolving through constant iteration. This approach allowed us to communicate freely with all team members and clarify our doubts.

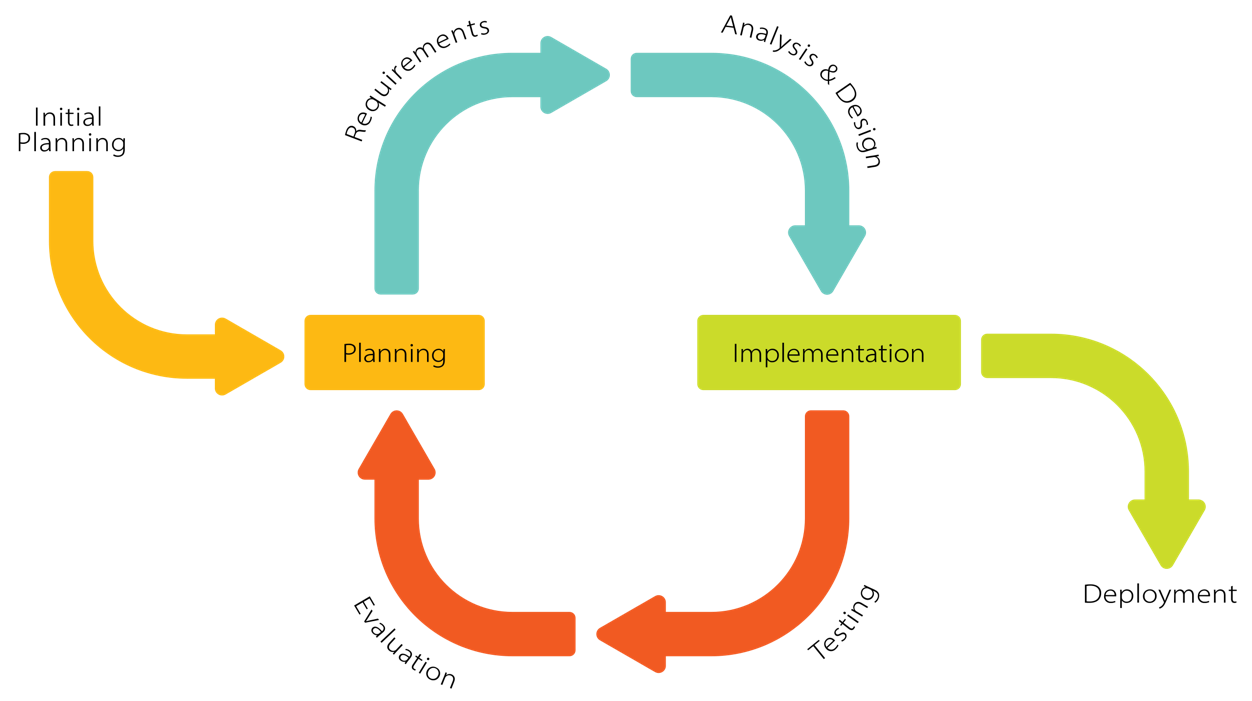


Figure 1: Iterative development model (by Krupadeluxe, wikipedia.org)

### Methodologies and technologies

Our project is a fully independent app as SaaS (Software as a Service). Users do not have to worry about data, O/S, servers, or storage – it is all adapted to work as a final product for the user site. We managed all of them and packed them into a single product.

We used Python language to program the backend (using OpenCV library for reading poses) and XXX for the frontend. Testing did not require any units or libraries; we coped with that manually. For now, we have decided not to use external databases and virtualization.

### Strategy

**Must have:**

ability to choose and show the video, correct assessing and scoring, showing the score, zero-delay playback

**Should have:**

showing the score after each move, precise scoring, zero-delay reading poses

**Could have:**

uploading dances by user, scoreboard, saving scores in the databases, custom background color, zero-delay scoring, safety contracts, YouTube connection

**Won't have:**

multiplayer, availability for smartphones or tablets

### The architecture

### Use cases

### Sequence diagram

### Functional requirements

1. The user must be able to choose a dance video.
2. The user must be able to watch a dance video.
3. The user must have access to all uploaded videos.
4. The system must generate a score.
5. The student should be able to dance without system interruptions.
6. The system must be guaranteed full Internet access.

### Non-functional requirements

**Product requirements:**

1. The system must be available for 22 hours daily for all users.
2. The system can have up to 2 technical breaks of less or equal to 1 hour daily.
3. The system must not have a latency greater than 60 seconds.

**External requirements:**

1. The system must comply with the rules of preserving users' privacy following the GDPR.
2. The system must meet the civil code and law requirements on copyright.

### Natural language specification

The web application must be easy to use for each type of user. The interface must be straightforward so that no instruction is needed to handle the program correctly. The system must allow users to dance to their camera and assess that performance. Dance must be adequately scored after each session. Users can choose as many dances as they want and use the system for as long as it is up.

### Story

### Scenario

### System context

# Designing

### Back-end

1. jak wygladal za pierwszym podejsciem + opis

Obraz zawierający tekst, zrzut ekranu, linia, diagram

Opis wygenerowany automatycznie

2. jak wygladal za drugim podejsciem + opis

3. wyjasnienie kodu

### Scoring system

1. pomysly + ostateczne wykonania

# Implementation

### Front-end

1. jak sie zmienial

2. wyjasnienie kodu

# Testing

An indispensable feature of incremental development is frequent testing. We started testing after the first implementation of the backend code. It showed essential flaws in program operation at the first stage – the laptop camera was not reading the user movement correctly. Early testing allowed us to revise and repair the code before merging it with the frontend part, which would cost much time in waterfall development. From that moment, tests showed compatibility between the requirements and the code. We could focus on accurate scoring, different testing environments, and program efficiency.

1. co pomogly zmienic testy

2. jak testowano system operacyjny

### Choreographies

1. wyjasnienie jakie sa

2. jakie pomysly na testy

### Verification

### Validation

# Maintenance

1. jak zamierzamy zajac sie programem po wypuszczeniu na rynek

# Overall output

1. jak finalnie wyglada program

# Challenges

1. jakie napotkano proeblemy ktore udalo sie rozwiazac

# Unresolved issues

1. co poszlo nie tak ze wydajemy z bledami

# Future improvements

1. co zamierzamy dodac w przyszlosci

2. jak czesto updaty jesli wgle

# What is next?

1. czy dajemy do sklepu

2. czy zamierzamy updatowac np zima

3. czy zamierzamy rozwiazac problemy

rozne systemy operacyjne opisz