Capstone Design Project 2

**Dance learning application utilizing pose estimation**

**Dance learning web application with pose estimation and deep learning**

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

December, 2023

**Kyungpook National University**

Table of contents

[Project description 3](#_Toc153052798)

[Defining requirements and specifications 3](#_Toc153052799)

[Features of a good software 3](#_Toc153052800)

[Development 3](#_Toc153052801)

[Technologies 4](#_Toc153052802)

[Strategy 4](#_Toc153052803)

[The architecture 4](#_Toc153052804)

[Use cases 5](#_Toc153052805)

[Sequence diagram 5](#_Toc153052806)

[Functional requirements 5](#_Toc153052807)

[Non-functional requirements 5](#_Toc153052808)

[Natural language specification 5](#_Toc153052809)

[Stories 5](#_Toc153052810)

[Scenario 6](#_Toc153052811)

[System context 6](#_Toc153052812)

[Designing 6](#_Toc153052813)

[Project's idea 6](#_Toc153052814)

[Back-end 6](#_Toc153052815)

[Scoring system 9](#_Toc153052816)

[Implementation 9](#_Toc153052817)

[Front-end 9](#_Toc153052818)

[Testing 9](#_Toc153052819)

[Choreographies 9](#_Toc153052820)

[Verification 10](#_Toc153052821)

[Validation 10](#_Toc153052822)

[End product, concerns and future releases 10](#_Toc153052823)

[Maintenance & future improvements 10](#_Toc153052824)

[Overall output 10](#_Toc153052825)

[Challenges 10](#_Toc153052826)

[What is next? 10](#_Toc153052827)

# 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.

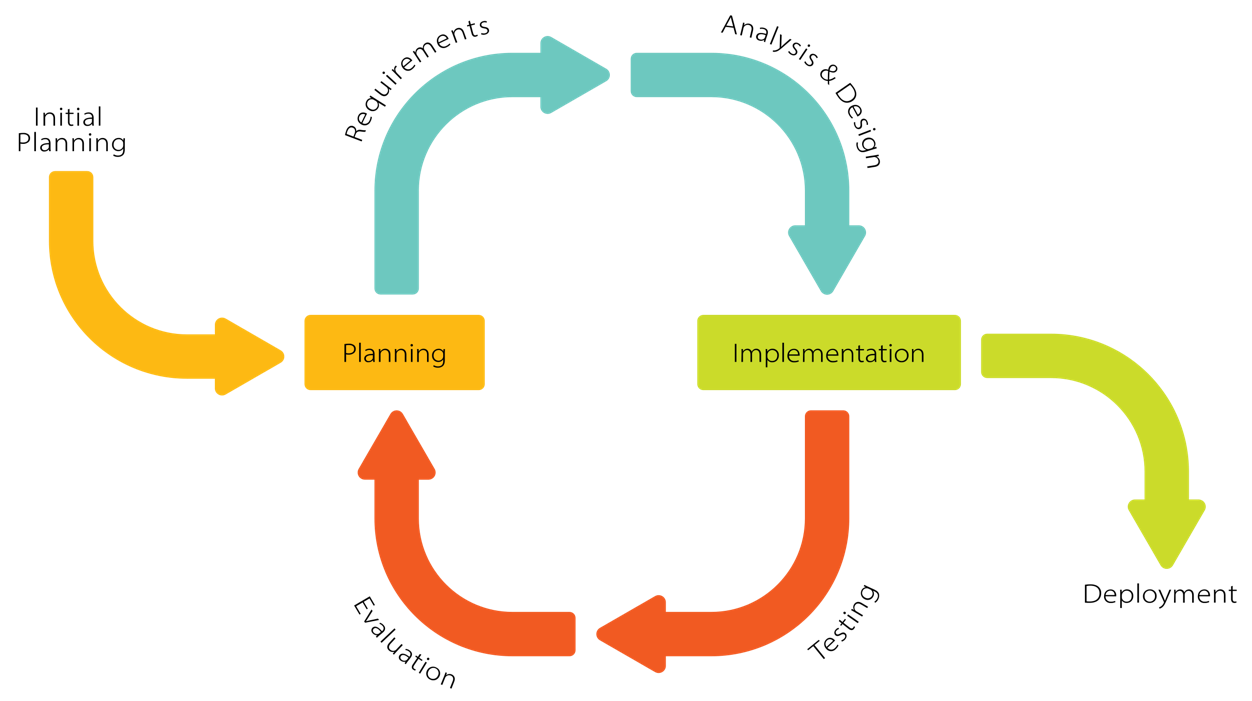


Figure 1: Iterative development model (by Krupadeluxe, wikipedia.org)

### 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 end product.

We used Python 3.10 language to program the back-end and Flask XXX with JavaScript for the front-end. Testing did not require any units or libraries; we coped with that manually.

### 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

**Back-end:**

landmarks (skeleton vertices) are the minor parts of the system; they, as groups, are stored in  
a container called raw skeleton; the raw skeleton is transformed into XXX, which is the highest point in the architecture tree

[doprecyzuj back-end Bartosza i dopisz Marcela]

**Front-end:**

### 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.

### Stories

Wiktoria is an exchange student at KNU. She and her friends are preparing an app for a course project. Her role in the project is to test the given output with dances. She can access the website application as a user – she can open, run, and try it by dancing to the laptop camera. After her performance, the app displays a score. She can assess her moves and, by that, check if the code works correctly. Her insights were possible thanks to a well-optimized program.

Alex is American and is keen on dancing. He enjoys learning new choreographies from YouTube services but finds them mostly unclarified. Alex would like to apply for dance classes, but instructors in his city do not teach choreographies to his favorite pop and hip-hop songs. He cannot afford a console to buy games such as Just Dance or Dance Central. His high school colleagues recommended him the dance learning application. Now, he can dance without spending money whenever he has time or willingness. What he likes the most about the app is that he can choose what videos to train.

Amelia is a dance instructor and a volunteer at the kids' hospital. She wanted a program to work with during her out-of-charge classes at the hospital. Amelia likes the convenience of the portability of the application and the variety of dances. She can easily access the website from the hospital using an Internet connection and her laptop. Amelia does not need any accounts or connected credit cards to provide smiles on the kids' faces. She uses the application weekly for the kids and sometimes plays with her friends for relaxation.

### Scenario

### System context

# Designing

### Project's idea

At first, the idea was to construct a drone (quadcopter), attach a single camera, and, by merging photos, create a 2D map aerial view of the KNU campus terrain. Then, the idea evolved to following a person or a car on the street. After a thorough discussion among the group members, we concluded that creating a dance learning application would be a more innovative and valuable project scope. The ability to create something new was crucial to us.

### Back-end

**Technology:**

The Python code, developed using version 3.10, combines OpenCV, MediaPipe, and CSV libraries. Matplotlib is utilized to visualize movements (solely for testing purposes). Bartosz explored alternative data models apart from Google MediaPipe but found limited accessibility and Internet access requirements. Drawing from MediaPipe documentation, Bartosz crafted specific sections of the program. The primary goal of the application was to implement an Artificial Intelligence (AI) model. This model is designed to detect body nodes from a single frame, making it particularly suitable for testing as it perceives and charts the body's nodes.

**Prototype:**

As a prototype, Bartosz created a code that calculated the skeleton vertices (*Figure 3*). He made a sample code showing the read skeleton movements and presented them as a video. In this version, each point had three coordinates – X and Y for two dimensions and Z for the third dimension. 3D could be added in future program updates since the code allows this improvement. Bartosz rewrote the code for 2D because it is enough to comprehend the application usage completely. He had to resign from multithreading because it did not comply with Grzegorz's idea on the front-end. One thread was supposed to read the user's body from the camera (and write its landmarks), and the other – created the skeleton from the given video.

Obraz zawierający tekst, zrzut ekranu, linia, diagram

Opis wygenerowany automatycznie

Figure 2: First version of the skeleton

**Final version:**

After discussing which points to choose, we focused on right and left hip, right and left knee, right and left ankle, right and left foot, right and left shoulder, right and left elbow, right and left wrist, right and left pinky finger, right and left index finger, and a nose (*Figure 3*). Every body part is essential in dancing, but depending on the genre, different points significantly impact overall performance. We did not differentiate based on dance styles, but the change in the nodes is possible.

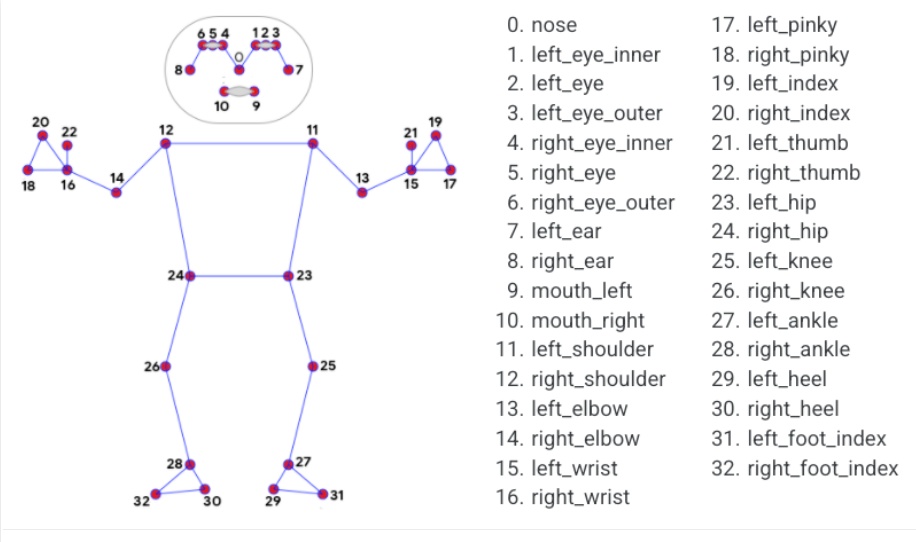


Figure 3: Available nodes on a skeleton

[fotka z odczytanymi punktami z video]

**Insights on the code:**

One of the essential low-level functions reads landmarks (points, nodes) from a single picture (and writes as a CSV file) based on the pixel list. It returns a data dictionary, which is later processed for a skeleton class as a skeleton. The data class of the landmark has X, Y, and  
Z coordinates (floats), an ID (for distinguishing nodes based on (*Figure 3*), and a name. It can be interpreted as a dictionary. Bartosz wrote a function that projects to the bool – when  
a node is not visible on the screen or in any other extreme scenario. To be consistent with other points, this created an empty skeleton with nullable landmarks, which can be compared by boolean means. The data class might be considered as an abstract class.

Another class creates a skeleton based on an image (representation of an abstract class). Another class connects not yet connected nodes, creates a graph relation (forest with one parent each), and normalizes points. Normalization is crucial since we will be measuring different people, so the program has to be accurate on lengths between the parent and the child points). It calculates the vector betwixt the parent and child, normalizes the vector on a picture, and then assigns it to the normalized parent's node. That is how the normalized node is created. Since each skeleton landmark requires a parent, there is a root object. Simply put, it is a skeleton without a parent and deprived of normalization. A skeleton is a container for landmarks (it enables accessing a landmark by its ID). It is an aggregation because the landmarks cannot exist solely (independently). Each skeleton has a timestamp assigned during reading from the video. It is needed for correct scoring.

The skeleton is divided into two types: collecting raw data from images, processing them into raw landmarks, and normalizing them to create the second ­– "fresh" skeleton. The variable deciding the parent-child weight can be changed to make different relations for other purposes. Another container is for fresh skeletons (with time stamps). Timestamps allow the withdrawal of a skeleton based on time (precision is based on rounding, so it is more of  
a closest-to-time skeleton). Dance manager class is mainly used to compare dances. It is  
a direct connection to the front-end which evokes this function. As input, it gets a given dance file (CSV) and creates a dance object. The web application decides from what time the video is played. It makes a ready final dance and an empty dance (for user input). In a loop, it reads a dance and goes through the process of landmark creation for the scoring system to be applied.

### Scoring system

1. pomysly + ostateczne wykonania

2. mock

# 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 back-end 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 front-end 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

# End product, concerns and future releases

### Maintenance & future improvements

If we commercialize the product and put it into the market, we will prepare for conservation. This includes regular software updates and patches to address any potential vulnerabilities or bugs that may arise. We will ensure that the end product meets the standards stated in this report requirements – with great emphasis on delays. We would prepare an immense dance library and create a score database with state-of-the-art security (prioritizing safeguarding user information).

Application extensions could include sign-up and sign-in options. We would allow users to use YouTube to the fullest because, currently, the application enables dancing to videos chosen by programmers. We would work on reducing delays and improving thorough scoring based on the newest algorithms and deeper testing. Another update could include 3-dimensional reading and assesion. This would require not only Bartosz's code update but also Marcel's. Finally, the multiplayer mode could be implemented alongside the online scoring board.

### Overall output

1. jak finalnie wyglada program

### Challenges

One of the first issues I encountered was a problem with running the application on macOS. It was not resolved before the back-end–front-end merge.

Since, at first, we were also writing Z coordinates for the 3D, Bartosz had to complement the code with a 2D version implementation. Even though the code was correct, it still used Y as  
a valid, not null, coordinate. Marcel's scoring system was faulty because of this. Bartosz encountered this error when explaining his code for the report purposes. After that, he resolved the issue, and the scoring ran correctly.

### What is next?

We could commercialize the app by adding ads to the web version. We could also write the applications for a desktop for Windows (still in Python) and macOS (in Swift). This would require another approach to designing and building the app but would allow more accessible and planned earnings from ads or paid apps.

Since we were creating the app for the CAPSTONE project and for PETE course, we could work on it pod okiem profesora z pete to upgrade the app over the basic zalozony level.

3. dopisz Marcela

4. dopisz Grzegorza

5. dopisz siebie

6. dodaj zdj duzo

7. dodaj schematy

8. dopisz rozdzialy

9. dopisz wiecej tekstu do rozdzialu

10. napisz o spotkaniach z pete

11. opisz forme pracy moze

12. nazwy rozdzialow

13. struktura pliku

14. jaki deadline?

15. czemu nie pararel processing