PROJECT 1-2
PHASE 3

Group 17: Alexandra Afanasiuc, Anna Balañá Zemanaj, Jakub Bujak, Olaf Deckers, Przemysław Grudka, Jagoda Jakuczun and Nina Żórawska

PLAN OF THE PRESENTATION

1. Recap - Phase 1 and Phase 2
2. Research Questions
3. ODE Solvers Experiments
4. Rule-Based Bot vs AI Bot Experiments
5. A* Algorithm
6. A* Algorithm Experiments
7. GUI
8. Conclusion
9. Outlook

RECAP

- Phase 1
  - ODE solvers - Euler and Runge-Kutta 4th order method
- Phase 2
  - Implementing our ODE solvers into physics engine
  - Creating a golf simulation with water and sand pits
  - Creating the Rule Based bot and AI bot

RESEARCH QUESTIONS

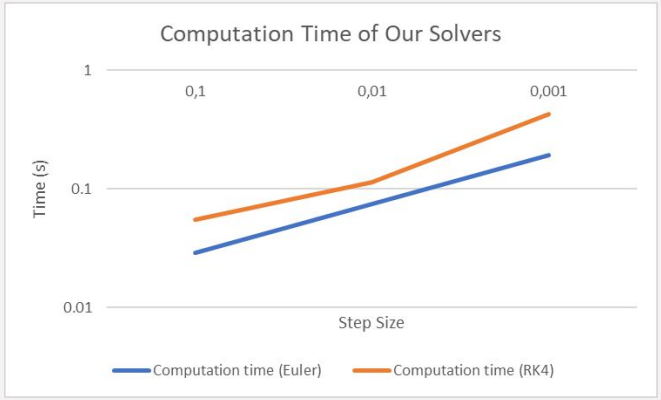
1. How can we accurately simulate the physics of a golf ball on a sloping course with obstacles using differential equations?

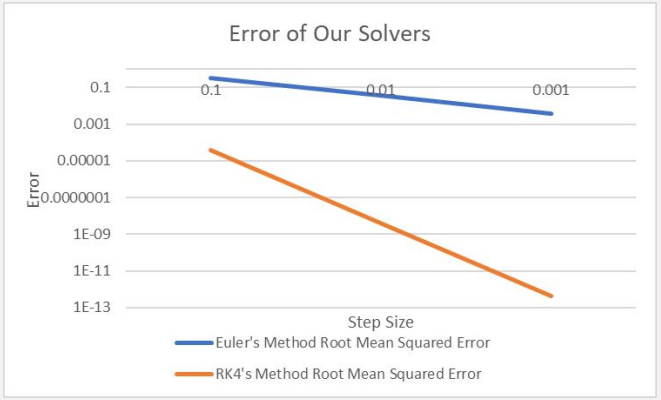
2. What are the optimal numerical methods for solving these ordinary differential equations in the context of this problem?

3. How can we design and implement an AI system capable of navigating the simulated courses to reach the goal in one shot and handling complex maze-like courses?

ODE SOLVERS EXPERIMENTS

Root Mean Squared Error and Computation Time of Euler and RK4 methods





System of differential equations used:

da/dt = b

db/dt = 6a - b

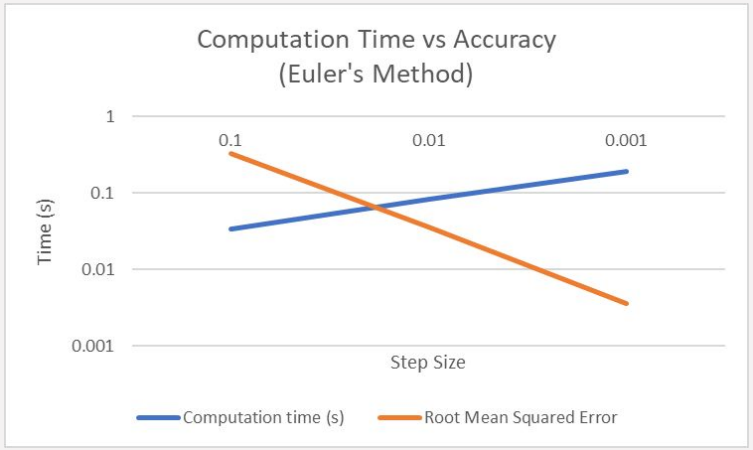
with initial conditions: a(0.0) = 1, b(0.0) = 0

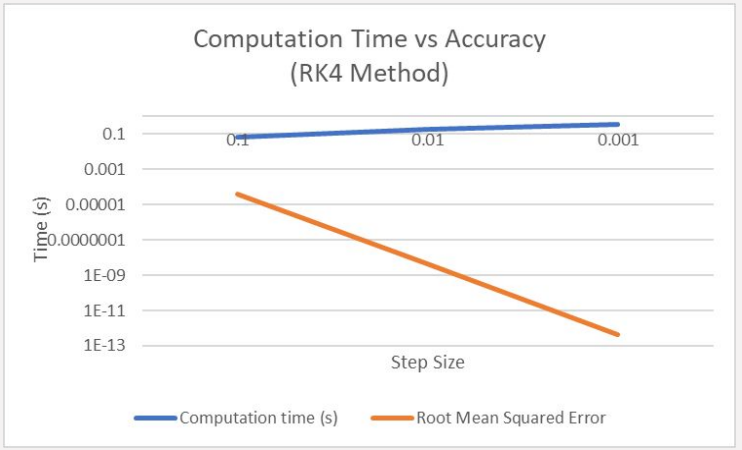
on the time interval t = [0.0,1.0]

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

ODE SOLVERS EXPERIMENTS

Computation Time vs RMSE for both Euler and RK4 methods





System of differential equations used:

$da/dt = b$

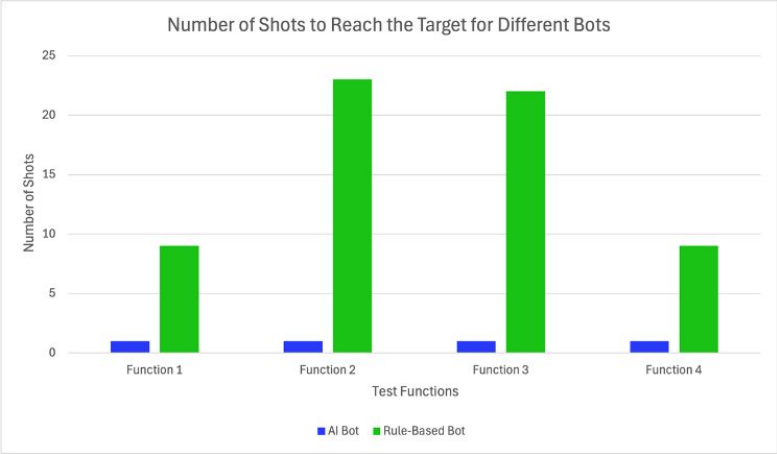
$db/dt = 6a - b$

with initial conditions: $a(0.0) = 1$, $b(0.0) = 0$

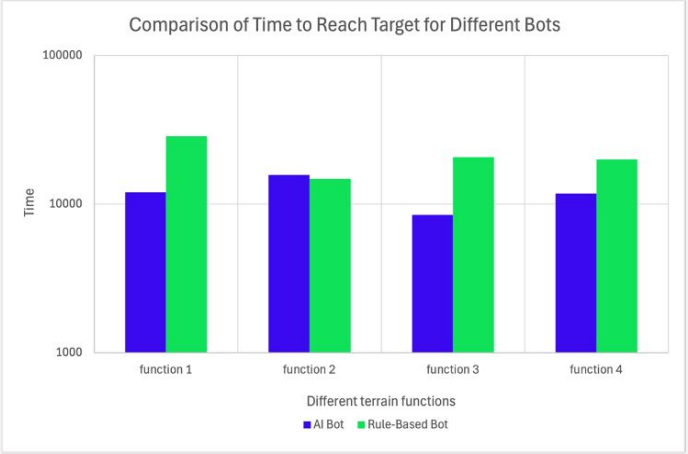
on the time interval $t = [0.0,1.0]$

AI BOT VS RULE-BASED BOT EXPERIMENTS

Number of Shots Taken to Reach Target



Comparison of Time to Reach Target



AI BOT VS RULE-BASED BOT EXPERIMENTS

Test conditions:

Test number	Terrain function	X coordinate target	Y coordinate target	Radius target	Initial position of the ball on the x-axis	Initial position of the ball on the y-axis
1	5	0	0	0.5	25	25
2	sqrt ((sin (0.1 * x) + cos (0.1 * y)) ^ 2) + 0.5 * sin (0.3 * x) * cos (0.3 * y)	4	1	0.5	15	20
3	0.4 * cos x + 10	0	0	0.5	10	10
4	0.4 * cos x + 0.6 * sin y + 10	0	0	0.5	10	10

AI BOT VS RULE-BASED BOT EXPERIMENTS

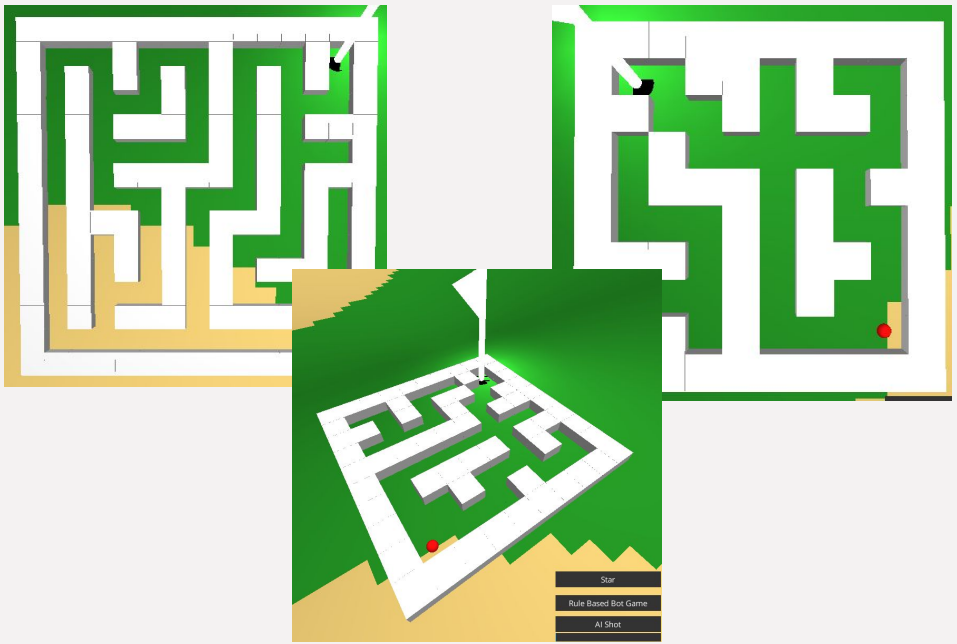
AI Bot vs Rule-Based Bot - Time to Reach Target from Different Distances

Exact Results for the bots for Function 2

Number of the Test	AI Bot Time in Milliseconds	Rule-Based Bot Time in Milliseconds
Test 1	13235	14787
Test 2	19905	-
Test 3	14104	-

function: “5”; target position: (0,0); target radius: 0.5;
initial ball coordinates: (10,10), (20,20), (30,30), (40,40)

CREATING MAZES



- The maze consists of walls stored in a list, defining its boundaries and pathways
- A 2D array represents the maze's structure, with each element indicating a wall (1) or open space (0)
- A small buffer around walls accounts for minor position inaccuracies, ensuring reliable collision detection
- We handle detecting collisions between a golf ball and maze walls, and calculate the new velocity of the ball after it bounces off a wall.

A STAR SEARCH ALGORITHM FOR IMPROVED AI BOT

Purpose: Efficiently navigate maze-like courses

Mechanism: Combines actual cost from start to node and heuristic estimate to goal.

Heuristic Function: Prioritizes nodes closer to the goal

Process:

- Grid of nodes representing the course.
- Select node with the lowest f(n)f(n)f(n) from the open set.
- Examine and update neighbors.
- Reconstruct path from goal to start once the target is reached

$$f(n) = g(n) + h(n)$$

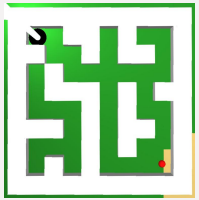
Cost Function

$$d_{\mathrm{T}}(\mathbf{p}, \mathbf{q}) = \|\mathbf{p} - \mathbf{q}\|_{\mathrm{T}} = \sum_{i=1}^{n} |p_i - q_i|$$

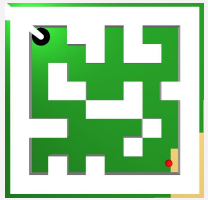
Manhattan Distance
(Heuristic Function)

A-STAR ALGORITHM EXPERIMENTS

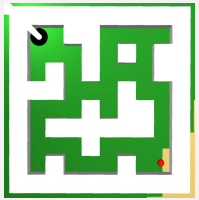
A Star Algorithm - Time to Reach Target in Different Maze



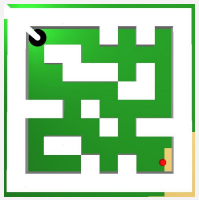
1



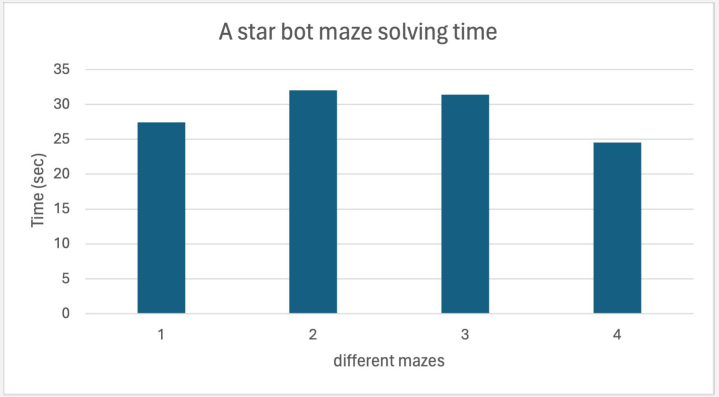
2



3



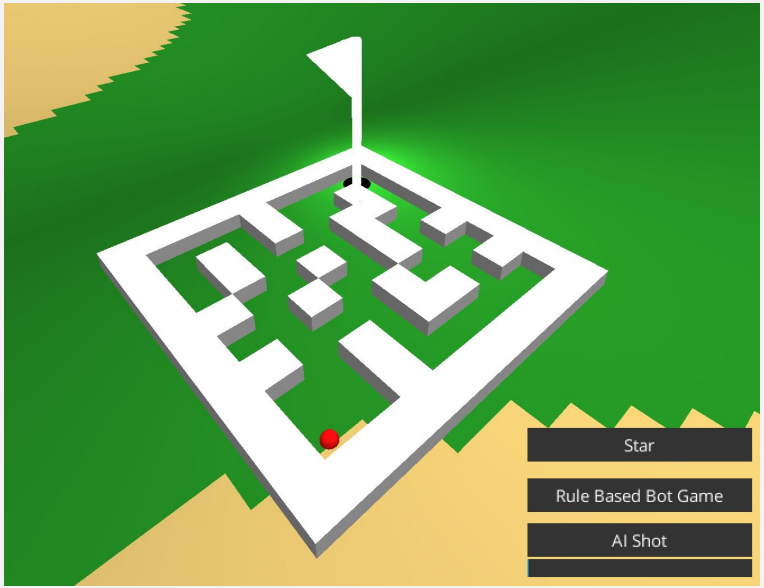
4

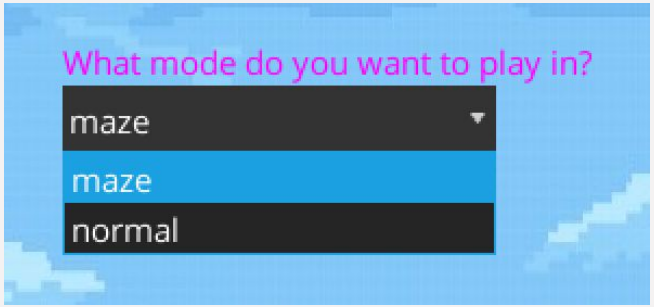


Performance of Different Bots on Maze-Like Courses

Maze	Rule Based Bot	AI Bot	A Star Bot
1.	Fail	Fail	Win
2.	Fail	Fail	Win
3.	Fail	Fail	Win
4.	Win	Fail	Win

GUI


Star
Rule Based Bot Game
AI Shot


What mode do you want to play in?
maze
maze
normal

CONCLUSION

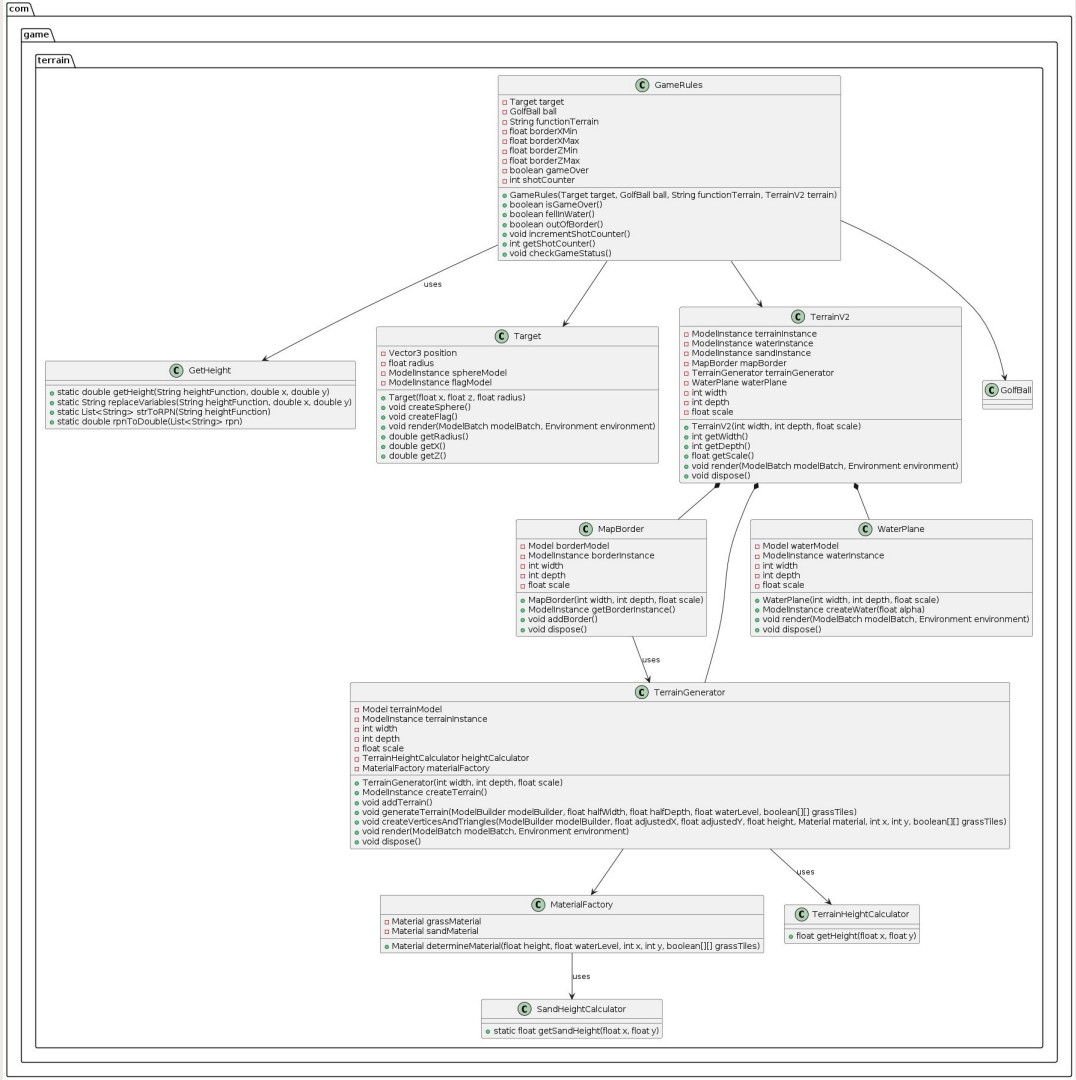
- RK4 exceeds in accuracy over the Euler method, but requires more computational time.
- AI bots that use advanced algorithms like A* significantly outperform simple Rule-Based bots.
- AI methods can be extremely useful for designing autonomous decision-making systems.

OUTLOOK

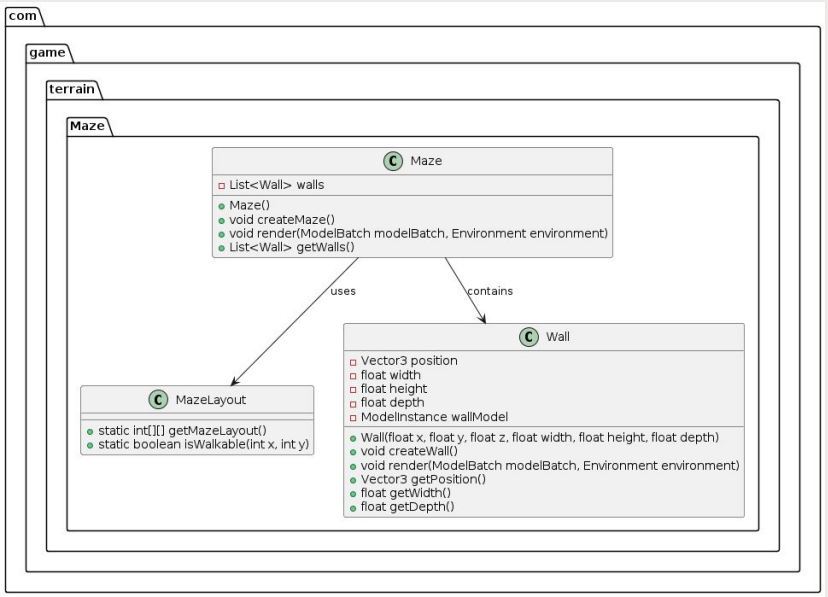
- Expanding our simulation to incorporate more complex environmental factors and real-world physics
- Enhancement of the RK4 method
- Development of an approach to combine RK4’s accuracy and Euler’s speed
- Further development of the AI bots

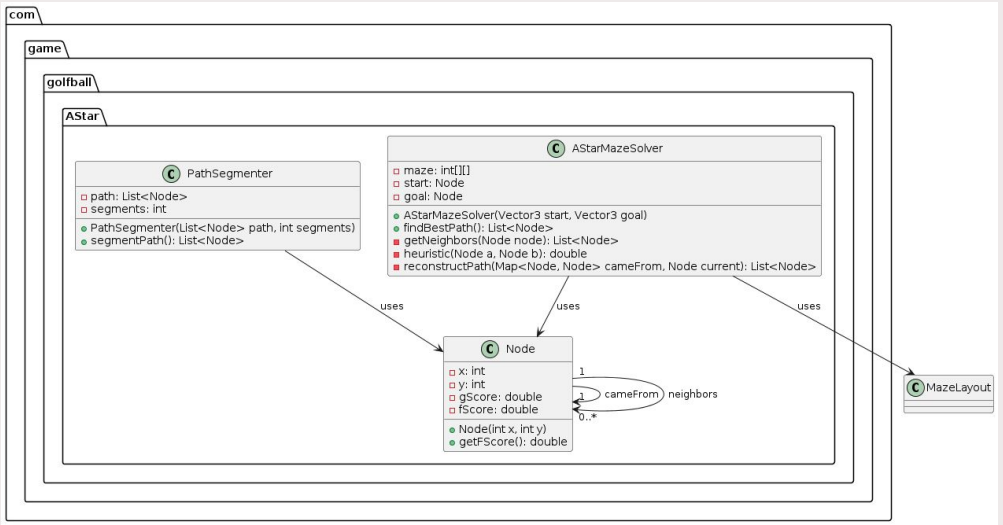
THANK YOU FOR YOUR ATTENTION

UML DIAGRAMS - APPENDIX



UML DIAGRAMS - APPENDIX





OVERVIEW

Student name	Coding Tasks Phase 3
Nina	GUI (80%), Experiments (25%)
Kuba	GUI (20%), Experiments (25%)
Jagoda	Correcting Runge Kutta solver (50%), Experiments (25%)
Olaf	UML Model(100%), Refactoring(100%)
Anna	Correcting Runge Kutta solver (50%), Experiments (25%)
Alexa	Maze (50%), A* Algorithm (50%)
Przemysław	Maze (50%), A* Algorithm (50%)

OVERVIEW

Highlevel Task	
Coding	Jagoda(14%), Przemek(25%), Nina(10%), Jakub(10%), Alexa(25%), Olaf(10%), Anna(6%)
Presentation	Nina(14%), Jagoda(14%), Anna(14%), Alexa(14%), Przemysław(14%), Jakub(14%), Olaf(14%)
Report	Anna(25%),Nina(5%), Jagoda(25%), Jakub(5%), Przemysław(5%), Alexa(5%), Olaf(30%)
Project management tasks	Jagoda(20%), Nina(20%), Jakub(12%), Anna(12%), Przemysław(12%), Olaf(12%), Alexa(12%)