Technical Report - **Product specification**

PASSIT

Course:	IES - Introdução à Engenharia de Software
Date:	Aveiro, 3rd January 2023
Students:	102442: Daniel Ferreira 103668: Gonçalo Silva 103600: Guilherme Antunes 103541: Pedro Rasinhas
Project abstract:	PASSIT is an integrated system that allows for the capturing, tracking and storing of various performance variables of football players during the course of a match. PASSIT allows for the persistent permeation of data as well as long-term tracking of said data.

Table of contents:

1 Introduction

The **goal** for this project was to propose, conceptualise, and implement a multi-layer, enterprise-class application.

With this in mind, **PASSIT** was developed: a web app that allows coaching staff of football teams to monitor, analyse and track various performance indicators of their teams as a whole and as individual players throughout matches.

2 Product concept

Vision statement

PASSIT will be used by head coaches and supporting staff alike, to centralise previously scattered data, in order to optimise game strategies, adjust player usage and compare and contrast with other teams.

It is expected that this app will be used in high-level competition scenarios, but due to the accessible nature of the system, having usage permeate to more amateur levels of the game would be a desirable and feasible outcome.

Personas

John Smith is a 30 year old assistant coach for C.D. Tondela.
Since his team doesn't have the largest financial capability in its league, he's always trying to find ways to drive efficiencies in his team through the use of simple, inexpensive technology. He does long hours at the club, so this goes hand in hand with his distaste for inefficiency since he's also constantly looking for ways to automate parts of his job and maybe become head coach of the team.

Motivation: John likes to keep track of players stats to make sure they're improving and which players are more suitable to implement certain tactics, not to mention being able to optimise the training given to players.

Mark Evans is a 19 year old rising star fresh out of Benfica's training grounds. Since he was a child, he dreams about being a name synonymous with football.
He had a difficult time from the beginning: he received an asthma diagnosis at an early age. Due to his condition, he had to train his stamina.

Motivation: He would like to be self-aware of his physical condition and his performances in games along the season hoping he can get the awards he deserves.

Olivia Jones is a 40 year old woman with one son. She's married to an avid football fan. Thanks to her husband, she started to follow the sport and developed a strong liking to it. Little by little, she started learning about the teams and the players.
These days, due to her job and having to take care of her young son, she doesn't have the time to watch as many games as she would like to, and therefore is generally not up to date on how her favourite players are performing.

Motivation: She feels the need to use PASSIT to check her favourite players PASSIT score after each game to see if they improved or not.

Main scenarios

John wants to implement a new tactic: John wants to try to transition his team into a 4-3-3 formation, but in order to do this, he needs to find out who on his team has the most stamina. To do this, he uses PASSIT to record data from a practice game and finds his top 3 less fatigued players to help spearhead his groundbreaking tactic.

John wants to make sure a player has a good recovery: Recently one of John's players was injured in a car accident and had to be under ventilation. In one of his first games since, John uses PASSIT to make sure his vitals and in-game performance are within a normal, admissible range.

Mark wants to see if he's improving: Even though he has asthma, Mark doesn't want it to stop him from achieving his dreams, so keeps giving everything he has in every practice session. That being said, he easily finds his limits, so he uses PASSIT to check not only if his vitals are within a safe range but also if they are improving the more he practises.

Olivia wants to know how her favourite players are doing: Due to a family dinner, Olivia and her husband couldn't watch their team's game, so they use PASSIT to see how well their favourite players did in that particular match.

Olivia wants to save players on her favourites list: After watching the World Cup Olivia discovered some players that she likes so for her to be up to date with the players performance she adds him to her favourites list.

Olivia wants to remove players from her favourites list: Due to recent controversies Olivia decided to stop following a player with such enthusiasm so she decided to remove him from her favourites list.

3 Architecture notebook

Key requirements and constraints

PASSIT is a web app, therefore it has a set of particular requirements and constraints, such as:

- All users must be correctly sorted on login, having the correct permissions and access to their data, not being able, therefore, to access sensitive data from other users.

- Our system should be able to generate data (simulating cyber-physical sensors) about every player's stats (heartbeat, stamina, velocity) in order to keep track of the players physical conditions, allowing the coach to manage the squad accordingly. Note that the player's stats are associated with a team, this being confidential, as said above, and only visible to the allowed personnel.

- According to that data, the system should be able to generate reports about the players conditions (players on top physical form, …). PASSIT should also be able to generate reports on the players evolution (players that improved the most, …).

- Our system should be able to generate a PASSIT score that shows the player's rating in a certain game, and, eventually, calculate a player's PASSIT average score (taking into account the PASSIT scores from all games). Note that this score is public and can be seen by everyone.

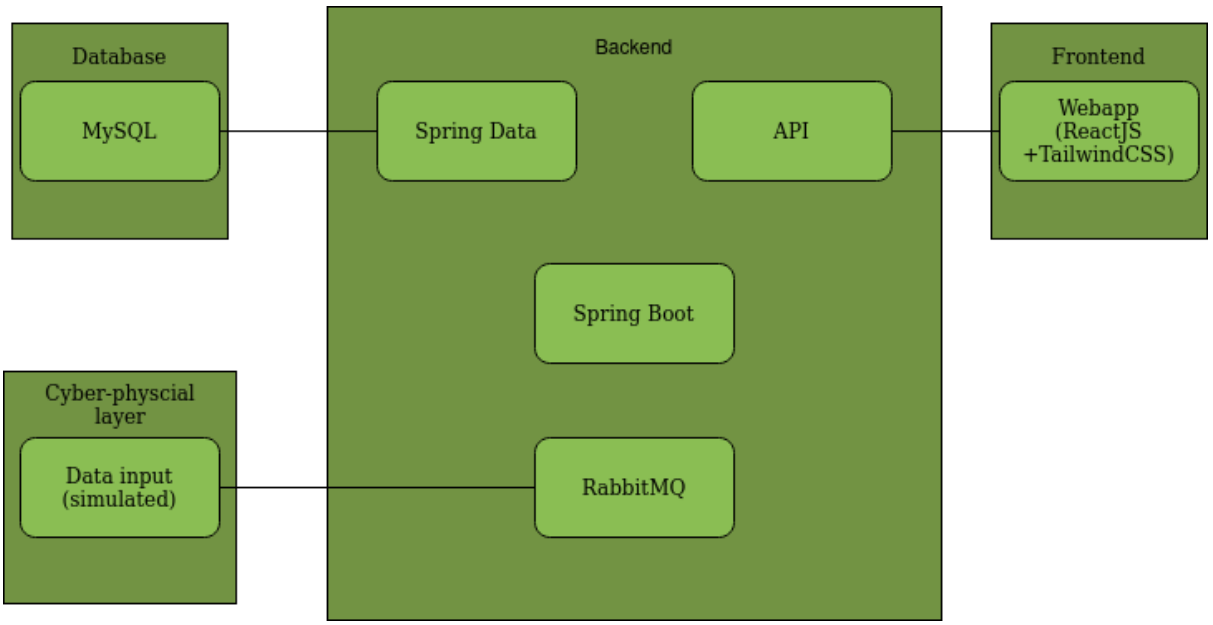
- Our system should be able to not only see the real-time generated data, but also to store it, so that the users can access that data in the future.

- Admins should be able to access all CRUD operations, while end users shouldn't be able to change or inject any data for statistical purposes, only being able to alter personal, inherent information.

- End users must be able to view data sorted and arranged in a multitude of ways, including receiving relevant suggestions based on context.

Architectural view

The system will be divided into three parts: a persistent database created in MySQL to store each teams' intrinsic information as well as the data generated from the app, a frontend web app developed in ReactJS with TailwindCSS that will interact with an API build in-house in order to manipulate data and a message queue implemented with RabbitMQ that will interact with the simulated data from each player.

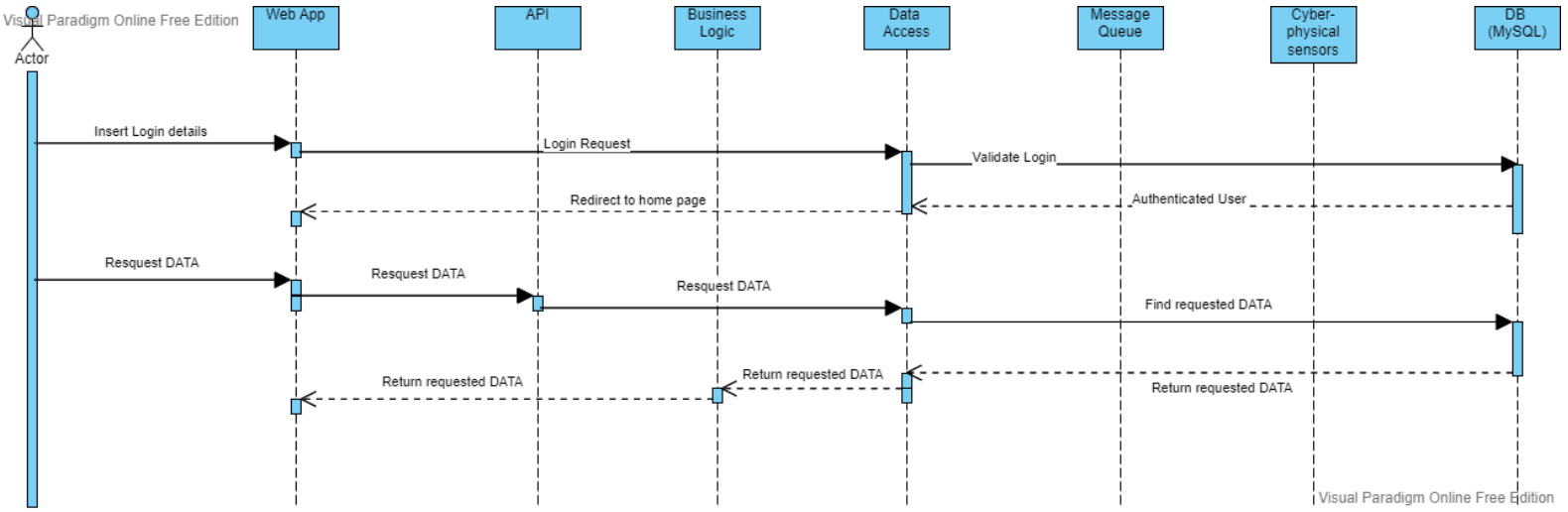


Module interactions

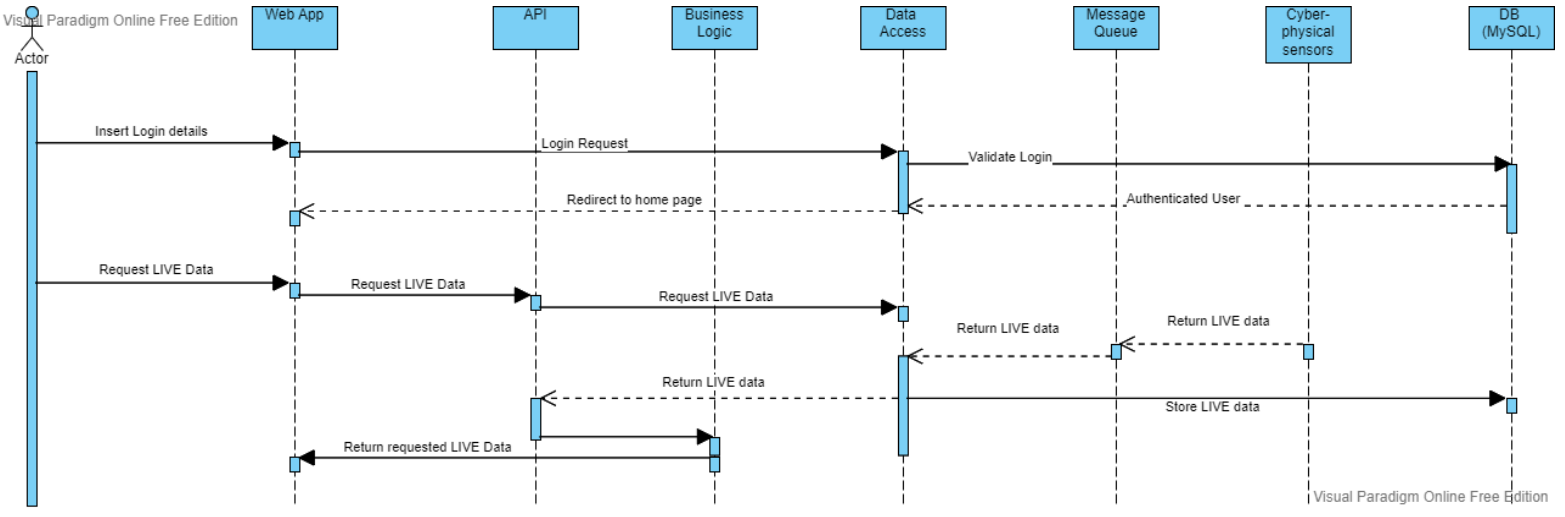
When a user opens the site and logs in, the Web App Module will interact with the Data Access Module which will interact with the DB Module to check if the login information is valid or not. This procedure that i just explained is

Interaction examples between the modules discussed above:

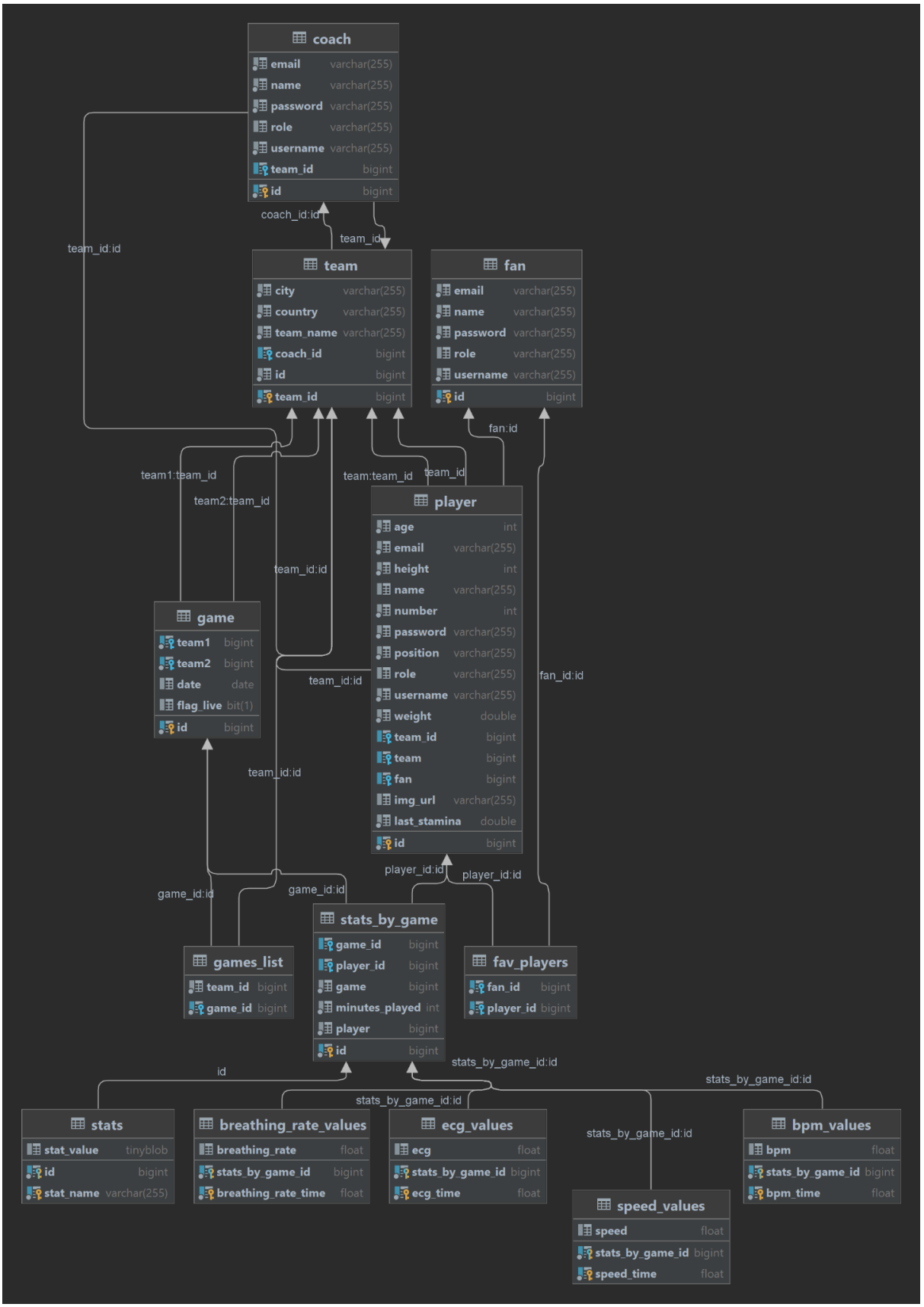
1. Access to persistent data, already in the database:
   -Once authenticated, the user can access the persistent data in the database, then the Web App Module sends a request to the API which will forward the request to the Data Access Module which will in turn find the information in the DB.



2. Access to live data, retrieved by the cyber-physical sensors:
   -Contrary to the previous example, the Data Access Module receives the data from the Cyber-physical sensors and the Data access model has the task of sending this data to the DB to store and also send it to the API.



4 Information perspective



5 API Documentation

SystemController

GET - Get Login Result - http://localhost:8080/api/v1/system/{username}/{password} ("username" being the username from the player, coach or fan and "password" being the password)

CoachController

GET - Get Coach - http://localhost:8080/api/v1/coach/{id} ("id" being the id from the coach)

GET - Get All Coaches - http://localhost:8080/api/v1/coach/all

POST - Add Coach - http://localhost:8080/api/v1/coach/add

FanController

GET - Get All Fans - http://localhost:8080/api/v1/fan/all

GET - Get Fan - http://localhost:8080/api/v1/fan/{id} ("id" being the id from the fan)

POST - Add Fan - http://localhost:8080/api/v1/fan/add

POST - Add Favorite Player - http://localhost:8080/api/v1/fan/addFavoritePlayer/{fan}/{player}
("fan" being id of the fan and "player" the id of the player to add)

POST - Remove Favorite Player - http://localhost:8080/api/v1/fan/removeFavoritePlayer/{fan}/{player}
("fan" being id of the fan and "player" the id of the player to add)

PlayerController

GET - Get All Players - http://localhost:8080/api/v1/player/all

GET - Get Player - http://localhost:8080/api/v1/player/{id} ("id" being the id from the player)

GET - Get Last Game - http://localhost:8080/api/v1/player/lastgame/{id} ("id" being the id from the player)

GET - Get the average Player statistics per game -
http://localhost:8080/api/v1/player/stats_user_game/{id}/{game}
("id" being the id from the player and "game" the id from the game)

GET - Get the average statistics of the StatsByGame -
http://localhost:8080/api/v1/player/stats_user_game/{id}
("id" being the id from the stats_by_game)

POST - Add Player - http://localhost:8080/api/v1/player/add

PUT - Update Players's Stamina - http://localhost:8080/api/v1/player/remstamina/{id}
("id" being the id from the player)

GameController

GET - Get Game - http://localhost:8080/api/v1/game/{id} ("id" being the id from the Game)

GET - Get All Games - http://localhost:8080/api/v1/game/all

GET - Get Players of Game - http://localhost:8080/api/v1/game/players/{id}
("id" being the id from the Game)

POST - Add Game - http://localhost:8080/api/v1/game/add

TeamController

GET - Get Team - http://localhost:8080/api/v1/team/{id} ("id" being the id from the Team)

GET - Get All Teams - http://localhost:8080/api/v1/team/all

GET - Get Players from Team - http://localhost:8080/api/v1/team/{id}/players
("id" being the id from the Team)

GET - Get Average Team Statistics per Game -
http://localhost:8080/api/v1/team/stats_team_game/{id}/{game}
("id" being the id from the Team and "game" the id from the game)

GET - Get Team's Highest Stats Players Per Game -
http://localhost:8080/api/v1/team//highest_player_stat/{id}/{game}
("id" being the id from the Team and "game" being the id from the Game)

POST - Add Team - http://localhost:8080/api/v1/team/add

StatsByGameController

GET - Get All StatsByGame - http://localhost:8080/api/v1/statsbygame/all

GET - Get Stats of Player - http://localhost:8080/api/v1/statsbygame/getstats/{id}
("id" being the id from the Player)

GET - Get Live Stats of Player from a Game -
http://localhost:8080/api/v1/statsbygame/live/getstats/{id}/{game}
("id" being the id from the Player and "game" being id from the game)

POST - Add StatsByGame - http://localhost:8080/api/v1/statsbygame/add

PUT- Update Minutes - http://localhost:8080/api/v1/statsbygame/minutesplayed/{id}
("id" being the id from the StatsByGamer)

PUT - Set the Individual Stats Data - http://localhost:8080/api/v1/statsbygame/addstats/{id}
("id" being the id from the StatsByGame)

PUT - Add Live Data to StatsByGame - http://localhost:8080/api/v1/statsbygame/live/addstats/{id}
("id" being the id from the StatsByGame)

6 Conclusion

In conclusion, the development of PASSIT has provided valuable lessons in the design and implementation of a multi-layer, enterprise-class application, including the use of microservices. Through the process of conceptualising and building the app, we learned the importance of identifying and defining the target audience and their motivations for using the app. We also gained an understanding of the importance of a clear and well-defined architecture, including the use of microservices, a frontend, API, and database, in creating a scalable and flexible application. The comprehensive API documentation also proved to be an essential aspect of the development process. Overall, the creation of PASSIT has been a valuable learning experience and has resulted in a useful tool for coaches and teams to optimise performance and gain a competitive edge in football.

7 References and resources

https://reactjs.org/docs/getting-started.html
https://v2.tailwindcss.com/docs
https://www.baeldung.com/spring-core-annotations
https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/
https://dev.mysql.com/doc/
https://docs.python.org/3/
https://www.rabbitmq.com/documentation.html

Regarding the frontend of the project, the ReactJs and Tailwind CSS documentation was used to aid with the website development.

During the development of the models we used the Spring Boot documentation and secondary sites about the annotations to make sure we were using the correct ones.

For medical concepts and data generation values (such as ECG, heart rate, respiratory rate and velocity), various sources were used such as various applications such as Samsung Health and multiple graph generation apps such as Geogebra,Photomath and Symbolab. There was also a book called "ECG-Assessment and Interpretation,Bradford C. Lipman" that helped to make a more accurate algorithm.
All of the code developed for the data generation was coded using Python 3.11 .

Another important technology was RabbitMQ being used as an intermediary between the data source and the backend(database) being also used to redirect the information to the queue.