PizzaDronz Project Requirements

This document will involve the software requirements for the software being tested for PizzaDronz app. Document will include *the stakeholders, the functional requirements, and measurable attributes of the software*.

Stakeholders:

This section lists all the stakeholders involved in the PizzaDronz. A stakeholder is someone who is directly influenced by our PizzaDronz app. Mentioning stakeholders is important since we will be priotising our requirements according to them. The following is a list of pertinent stakeholders:

Students: The primary users of the PizzaDronz app are the students, who utilize it to place orders at various restaurants and have their pizzas delivered by drone.

Public (Local Populance): We must ensure that the drones do not endanger public or environmental safety because they will be operating in public places. Additionally, we must make sure that the environment and general population are not in danger from the drones.

Restaurant owners: The system's service suppliers are restaurant owners. Therefore, the menus that each pizzeria offers and all of the restaurants are represented by these restaurant owners. Additionally, they enable the drones to place and then pick up the orders.

Software engineers: Delivering the finished product is the responsibility of this stakeholder, which is me in this project. Because they are directly involved in implementing all the software required to make the PizzaDronz app completely functioning, software designers are a relevant stakeholder.

Drone suppliers: These stand for a different business that will make the drone. They are a pertinent stakeholder because our PizzaDronz app depends on the drone, which will be connected with the delivery algorithm to facilitate the delivery of as many orders as feasible in a single day

Government: The government would play an important role as a stakeholder in a number of ways. They would have to control the service because it must comply with applicable air traffic regulations and not interfere with inhabitants' daily lives. In order to turn a profit, the government could also provide grants.

Measurable attributes:

We list all of the PizzaDronz app's measurable attributes in this section, along with any extra details that may be required for better understanding. The following is a list of measurable attributes:

Performance: The system's average number of pizza orders delivered in a given day can be utilized to measure its performance.

Fairness: If at least a specific amount of orders from each restaurant on the app are delivered on a given day (i.e., no single restaurant from which insufficient orders are delivered), then the system is considered to be fair

Efficiency: The speed at which the drone's flight path is planned and the output files are generated can be utilized for measuring the system's efficiency.

Usability: The system's operation can be measured by how simple it is for customers to use our product.

Attractiveness: How appreciated and valued the system is by end users can be utilized to measure how attractive the system is.

Reliability: The quantity of errors that occur within a specified time frame (such as a day) can be utilized for measuring the system's reliability.

Recoverability: The time it takes for a system to function normally after a failure can be utilized to measure its recoverability.

Maintainability: The simplicity of adding new features and the time it takes to restore the system to its previous state of complete functionality are two indicators of a system's maintainability.

Functionality: the functionality of the system can be measured as how well the system has met all the required requirements.

Security: The security of a system is determined by assessing the effectiveness of its measures to block unauthorized access, use, disclosure, or destruction of data.

Privacy: The measurement of privacy in a system considers the effectiveness of encryption for data in transit, reception, and storage, the locations of data storage, the duration for which data is retained, and the extent to which users are informed about the handling of their data.

Functional Requirements:

We provide all of the functional requirements that need verification for the PizzaDronz app in this part, along with any further information that may be relevant, including the category to which they apply. The definition of a functional requirement is "what the system should be able to do." I will also provide measurable attributes of each requirement according to our definition. The following is a list of the functional requirements:

1)System shall compute the distance between any two given locations using the Pythagorean theorem. **Functionality**

2) System shall not allow any point which is further away than the distance tolerance of 0.00015 degrees to be considered close. **Functionality**

3) System shall only allow the drone to fly or hover. **Functionality**

4) System shall only allow the drone to fly in one of the 16 directions specified in the coursework specification. **Functionality.**

5) System shall ensure that, once the drone enters the University’s Central Area, it does not leave this area until the current order is delivered. **Functionality**

6) System shall ensure that the drone never enters a no-fly-zone**. Functionality, reliability, and recoverability.**

7) System shall retrieve data about different components (such as University's Central Area, No-Fly-Zones, restaurant locations and order information) from a REST server. **Functionality and maintainability**

8) System shall allow the user to input 3 command line arguments, which are to be validated before the program runs. These 3 arguments are: a URL (to server as an address for the REST server), a date (to be used for retrieving all the order information for the day), and a word. **Functionality, reliability, and recoverability.**

9) System shall store the 10 possible outcomes for a given order. **Functionality.**

11) The system shall process and validate each order fetched from REST server to make sure its correctness and completeness. Details such as order number, order status, and the order validation code should be extracted from the REST server. **Functionality.**

-Order Validation must be based on these criterias

- Order number should be an 8-digit hexadecimal number.

-Credit card number must be a valid 16 digit number.

-Credit card expiry date must be after the order date.

-The CVV must be valid for the 3 digit number

-The order must contain between 1 and 4 pizzas inclusively.

-Pizzas in the order should match the pizza names on the restaurants' menus.

-All pizzas should be from the same restaurant:

-Total cost of the order must be equal to the sum of pizza prices plus the

fixed delivery fee of £1.

12) The application shall create 3 output files. **Functionality**

-A file which contains JSON records for each order on a given

date. File must include order number, the outcome associated

with the order and the total cost of the order.

-A file which contains JSON records representing moves that drone

makes on a given date. File must include the order number

being delivered, location of the drone before the move and after it (in terms

of longitude and latitude), angle of the move and the value in nanoseconds

representing the time elapsed since the start of computation.

-A file which includes a list of coordinates in GeoJSON format which illustrates

the flightpath of the drone

13) System shall generate a flight path for a singular drone to deliver a pizza order. **Functionality**

Non-Functional Requirements:

Performance

-The application shall not exceed a runtime of 60 seconds

Efficiency

- Number of delivered orders per day should be maximized as an efficiency metric. So that the application should be able to find an optimal

Security

-System needs to ensure that user data is protected

-System shall detect command line injection attacks by using input validation, sanitisation.

-System must use HTTPs connection with the Rest Service.

-System must make sure that external services have SSL certificates.

Qualitative Requirements

-System needs to make sure that safety and privacy of stakeholders are priority when operating.

-System implementation must be structured well and maintainable for upcoming modifications. (Maintainability)

-System should be able to handle errors properly providing feedback when exiting.

- The application should terminate and inform the user about the error if there exists an incorrect input argument in the application.

- If there exists an error occurring while application is executing, then they should be handled by terminating the application and user needs to be informed about the cause of the error.

Level of Requirements

System Level:

1)The application shall create 3 output files.

2)The system shall process and validate each order fetched from REST server to make sure its correctness and completeness. Details such as order number, order status, and the order validation code should be extracted from the REST server.

3)System shall allow the user to input 3 command line arguments, which are to be validated before the program runs. These 3 arguments are: a URL (to server as an address for the REST server), a date (to be used for retrieving all the order information for the day), and a word.

4)System shall generate a flight path for a singular drone to deliver a pizza order

5)System needs to make sure that safety and privacy stakeholders are priority when operating.

6)System implementation must be structured well and maintainable for upcoming modifications. (Maintainability)

7)System should be able to handle errors properly providing feedback when exiting.

8)The application should terminate and inform the user about the error if there exists an incorrect input argument in the application.

9)If there exists an error occurring while application is executing, then they should be handled by terminating the application and user needs to be informed about the cause of the error.

10)System needs to ensure that user data is protected

11)System should be able to detect command line injection attacks by using input validation, sanitisation.

12)System must use HTTPs connection with the Rest Service.

13)System must make sure that external services have SSL certificates.

14)The application shall not exceed a runtime of 60 seconds.

15)Number of delivered orders per day should be maximized as an efficiency metric. So that the application should be able to find an optimal.

Integration Level:

1)System shall retrieve data about different components (such as University's Central Area, No-Fly-Zones, restaurant locations and order information) from a REST server.

Unit Level:

1)System shall compute the distance between any two given locations using the Pythagorean theorem. **Functionality**

2) System shall not allow any point which is further away than the distance tolerance of 0.00015 degrees to be considered close. **Functionality**

3) System shall only allow the drone to fly or hover. **Functionality**

4) System shall only allow the drone to fly in one of the 16 directions specified in the coursework specification. **Functionality.**

5) System shall ensure that, once the drone enters the University's Central Area, it does not leave this area until the current order is delivered. **Functionality**

6) System shall ensure that the drone never enters a no-fly-zone**. Functionality, reliability, and recoverability.**

7)Order Validation must be base on:

- Order number should be an 8-digit hexadecimal number.

-Credit card number must be a valid 16 digit number.

-Credit card expiry date must be after the order date.

-The CVV must be valid for the 3 digit number

-The order must contain between 1 and 4 pizzas inclusively.

-Pizzas in the order should match the pizza names on the restaurants' menus.

-All pizzas should be from the same restaurant:

-Total cost of the order must be equal to the sum of pizza prices plus the fixed delivery fee of £1.

8)Output file creation must be based on these criterias:

-A file which contains JSON records for each order on a given date. File must include order number, the outcome associated with the order and the total cost of the order.

-A file which contains JSON records representing moves that drone makes on a given date. File must include the order number being delivered, location of the drone before the move and after it (in terms of longitude and latitude), angle of the move and the value in nanoseconds representing the time elapsed since the start of computation.

-A file which includes a list of coordinates in GeoJSON format which illustrates the flightpath of the drone