

Project Management Plan

*Indoor climate control system*

**CEO:** Brice Guayrin

**TUTOR:** Oswald Figaroa, Xuemei Pu

**TEAM MEMBERS** (roup #3):

### Victor Covalciuc Žana Bašić Sonam Lama Farros Ramzy Andy Verkooijen

List of Figures

[Terms & abbreviation 3](#_Toc97559710)

[1. Project description 4](#_Toc97559711)

[1.1 Context 4](#_Toc97559712)

[1.2 Project goal 4](#_Toc97559713)

[1.3 Project scope 4](#_Toc97559714)

[1.4 Research questions 5](#_Toc97559715)

[1.5 End products & deliverables 5](#_Toc97559716)

[2. Project Organisation 6](#_Toc97559717)

[2.1 Stakeholders and team roles & responsibilities 6](#_Toc97559718)

[2.2 Communication 7](#_Toc97559719)

[3. Activities and time plan 8](#_Toc97559720)

[3.1 Phase of the project 8](#_Toc97559721)

[3.2 Milestones 8](#_Toc97559722)

[4. Risk management 9](#_Toc97559723)

[5. Configuration Management 9](#_Toc97559724)

[5.1 Task Management Strategy 9](#_Toc97559725)

[5.2 System Merging Strategy 10](#_Toc97559726)

[5.3 Testing Strategies 10](#_Toc97559727)

[Reference 11](#_Toc97559728)

Document history

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Author | Date | Description | Status |
| 1.0 | Group #3 | 26-02-2022 | Draft Project Plan | Pending, WIP |
|  |  |  |  |  |

# Terms & abbreviation

|  |  |
| --- | --- |
| SDD | System Design Document |
| UI | User Interface |
| MVP | Minimum Valuable Product |

# Project description

## Context

Airios is a systems manufacturer, which produces custom-made electronics for residential climate control, that was founded in 1982. It began as a joint venture with Honeywell, although it is an independent organization today. Airios specializes in climate control systems but despite them being the strong suit of the company, Airios has also set their foot into other branches revolving around the control of the environment such as the connectivity, appearances and inner components of the products. The company strives to create its products like no other on the market by researching customers' needs as well as applying their feedback along the way. The final product is configured with the necessary safety measures as well as the reliability of the entire system.

## Project goal

The goal of this project is to produce a smart indoor climate control system, which will provide the user with requested contentment. The problems the company is being presented with are climate changes which cause the weather to be more unpredictable than ever along with the bad air quality. In this project Airios will attempt to create a system that will automatically regulate the indoor temperature by comparing it to the preferred user setting and adjusting accordingly. Additionally, the company will work on a ventilation system which will turn on fans and bring fresh air when the detected CO2 levels are too high in comparison to those of a desirable living environment.

## Project scope

|  |  |  |  |
| --- | --- | --- | --- |
| **Inside scope:** | | **Outside scope:** | |
| 1 | Ventilation Box (Simulated App) | 1 | Mobile App |
| 2 | Room Control UI | 2 | Scheduler |
| 3 | Wireless Communication | 3 | GPS Sensor |
| 4 | Database | 4 | Damage/Harm Detector |
| 5 | Analytic App | 5 | / |
| 6 | SDD | 6 | / |
| 7 | Room Climate Sensors (Temperature, Humidity, CO2, TVOC) | 7 | / |
| 8 | [Extra Assignments] | 8 | / |

## Research questions

Main:

How can modern advancements in technology such as new types of sensors and creative thinking can contribute to improving the control of the indoor climate ?

Subs:

1. What features can make the automatic system work properly?
2. What is required to make the system adapt to its environment climate?
3. How to make the system communication reliable?
4. How frequently should the communication protocol receive data?
5. How to make the smart home indoor climate control system reliable ?

## End products & deliverables

-Research about the impact of CO2

How does it impact our lives and what are the most common causes of excessive CO2 emission

# Project Organisation

## Stakeholders and team roles & responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Abbreviation** | **Role and functions** | **Availability** |
| *Contact name (and specify further detail as needed, e.g., email or Tel nr).* | *Abbreviation can help, e.g., when using the name in tools like Jira or MS project.* | *See above.* | *When is the person available for your project (define this in the way most relevant for your project, e.g., which days are available, the amount of time, or in which phase of the project).* |
| *Brice Guayrin* | *B. Guaryin* | *Product Owner of Airios Department* | *-Monday from 13 pm, Tuesday, Wednesday, Thursday, Friday*  *-Fontys R.10 or* [*b.guayrin@fontys.nl*](mailto:b.guayrin@fontys.nl) |
| *Victor Covalciuc* | *V. Covalciuc* | *Project Team Leader ( Scrum Master )* | *-Monday & Wednesday, 09.00 - 21.00*  *-Tuesday or Thursday, 09.00 - 21.00*  *-Friday 09:00 - 16:00*  *-Saturday 10:00 - 12:30*  *-Fontys R.10 or* [*v.covalciuc@student.fontys.nl*](mailto:v.covalciuc@student.fontys.nl) |
| *Žana Bašić* | *Z. Basic* | *Project Team Co-Leader*  *( Co - Scrum Master )* | *-Monday & Wednesday, 13.00 - 16.00*  *-Tuesday or Thursday, 16.00 - 17.00*  *-Friday & weekends 16.00 - 20.00*  *-Fontys R.10 or* [*z.basic@student.fontys.nl*](mailto:z.basic@student.fontys.nl) |
| *Farros Ramzy* | *F. Ramzy* | *Project Team Member* | *-Monday & Wednesday, 13.00 - 16.00*  *-Tuesday or Thursday, 16.00 - 17.00*  *-Fontys R.10 or* [*f.ramzy@student.fontys.nl*](mailto:f.ramzy@student.fontys.nl) |

|  |  |  |  |
| --- | --- | --- | --- |
| *Sonam Lama* | *S. Lama* | *Project Team Member* | *-Monday & Wednesday, 13.00 - 16.00*  *-Tuesday or Thursday, 16.00 - 17.00*  *-Fontys R.10 or* [*s.lama@student.fontys.nl*](mailto:s.lama@student.fontys.nl) |
| *Andy Verkooijen* | *A. Verkooijen* | *Project Team Member* | *-Monday & Wednesday, 13.00 - 16.00*  *-Tuesday or Thursday, 16.00 - 17.00*  *-Fontys R.10 or* [*a.verkooijen@student.fontys.nl*](mailto:a.verkooijen@student.fontys.nl) |

## Communication

Communication within the team and with the client is divided into several different meetings over different mediums as follows:

#### Daily stand-up:

Team meetings are held in person at the University during weekdays to discuss individual hurdles and task progress of each team member. Remote online meetings are organized via discord and Microsoft Teams during weekends.

#### Attendees:

-Team members

#### Bi-Weekly:

Sprint demo meeting with the client is scheduled every 2 weeks, through emails, depending on the availability of the client. A proof of concept of the previous sprint is demonstrated followed by backlogs for the next demo to be agreed on with the client.

#### Attendees:

-Team members

-Client

-Tutor

#### Occasional:

* Pre-demo meetings to prepare for the demo and prioritize backlogs to discuss with the client for the next sprint.
* Post demo meeting to discuss outcomes of the meeting and assign backlogs among team members.
* Remote meetings via Microsoft Teams and Discord server when needed.

#### Attendees:

-Team members

-Client

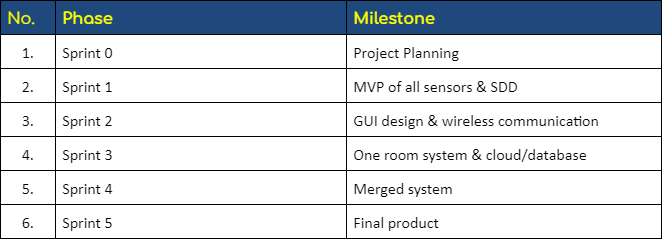
-Tutor

# Activities and time plan

## Phase of the project



## Milestones



# Risk management

|  |  |  |  |
| --- | --- | --- | --- |
| Risk | Probability | Impact | Countermeasures |
| Absent team members due to unforeseeable circumstances. E.g. sickness | HIGH | HIGH | - Distribute the remaining work package among team members.  - Git repository, daily stand-up meetings and Kanban board to keep track of assigned tasks and progress. |
| Technological defects  E.g. data loss, laptop breakdowns, and defective products. | MEDIUM | MEDIUM | - All code is backed-up in a GIT repository.  - Thoroughly test end products before demo or delivery. |
| Scope Creep E.g.  Additional requests of features from the clients existing outside the project scope. | HIGH | HIGH | - Well defined project scope outlining inside scope and out of scope tasks.  - Determine the feasibility of the requests and implement changes if possible, within the given time frame. |
| Failure to deliver a complete working end- product | HIGH | HIGH | - Prioritize important backlogs, highlight bottlenecks, and thorough product tests. |
| Absent team members due to unforeseeable circumstances. E.g. sickness | HIGH | HIGH | - Distribute the remaining work package among team members.  - Git repository, daily stand-up meetings and Kanban board to keep track of assigned tasks and progress. |

# Configuration Management

## Task Management Strategy

In this part, the team will split the features into some smaller tasks to work on. And each task should be splitted equally on their quantities, difficulties, and milestones. The team will use a scrum board to manage this task management per each sprint, and will use a repository to organize each update version of their tasks.

* Scrum Board

The scrum board will have some categories like “New”, “Active”, and “Closed” by its standard. And each of these categories has its own functionalities.

In the “New” category, all of the available tasks will be displayed. These are the free tasks that can be picked up by a team member to work on. In “Active”, the board will display every task that is currently being done. There is a developer working on each of their tasks labeled in this category. And if a task has been finished, that task can be just placed in the "Closed” category so everybody knows that the task is done.

However, the scrum board can be extended to have more than these three categories. For example, the team members could add a “Review” category between the “Active” and the “Closed” category, which means someone is still checking and figuring out if something has been missing in a task that was done before it is officially closed.

* Repository

The repository is a proper place to split, update, and merge a task to work for the industrial project. It is because of the safe use of it and the easy way to check or go back to the previous version of the task that has been published in it by multiple versions of its branch. To control each version update for every task, the team will use git for their repository.

There are two types of branches that the team will use. One of them is the dev branch (<task\_name>\_dev) where each member do their own tasks for their product features, and the other one is the master branch (<project\_name>\_master) where the team member should not touch until every smaller tasks on the active category finished and merged with each other.

## 5.2 System Merging Strategy

During working for this project per task, the team should merge every finished update until these updates become a complete product by the end of this project. There are two ways to do this merging strategy, which are:

* Merge to Dev

Any developer can merge their dev branches with each other only if these branches are included in the same user story or feature area. The developer must make sure that both of the dev branches which will be merged later are already reviewed before, and if they want to merge features or completed acceptance criterias of the user story, the dev branch in each task must be completed first.

* Merge to Master

This part of merging is only allowed when every feature is ready and merged into a one complete product, and that product is already tested before this merging is applied. It is because this merging will include every branch under the master branch into one completed repository.

## 5.3 Testing Strategies

Testing strategy is a way to ensure the quality of the product before releasing it. Testing strategy will include:

* Unit-testing

This testing basic approach is followed by the programmer to test the unit of the program. It helps the developers to know whether the individual unit of their code is working or not.

* Integration-testing

This type of testing focuses on the construction and the design of the product. The integrated units should be checked if they are working without any errors or bugs before doing this testing strategy.

* System-testing

This testing strategy checks the main functionality, security, and portability of the product before it goes to the hand of the customer. The product should be fully compiled and integration-tested before it is going into this testing part.

# Reference

# Airios Webpage

# <https://www.airios.eu/>

# 

# 