Faculty of Computers, Informatics and Microelectronics

Technical University of Moldova

AMSI

Course Work

Project description. Analysis and modelling of your project.

Authors:
Tanașciuc Macarie

Supervisor:
Radu Melnic

Chișinău, 2017

Contents

Introduction

The **Tracking System** is a multipurpose tracking system which can help in tracking the data on a computer.This can be used in various ways like History based tracking in case the user had an important work that he would want to remember each step done for offline working routine, or in case of an organization to supervise the work of its employees. When the soft is initialized it opens up in offline mode and has interface methods to extend to an online mode which can add the ID's of the other users in order to create an Admin Supervisor.The Admin supervisor has control over all the options in whole group. The Interface of the Tracking system contains various additional options in order to be able to create new tracking methods and filter the final data in form of Report for a filtered method. The filtered method can create or import methods for data filtering. The tracking methods can be chosen in a checkbox form.The default tracking methods will contain Keyboard type tracking,Mouse tracking,Upload/Download data track and a low quality(for fast data transmition) screenshot during a chosen time interval and/or activity.More tracking methods can be further added in the system, in of themselves the tracking methods are mini systems made to work on specific tasks.

1 Model of the application and Use Case Diagrams.

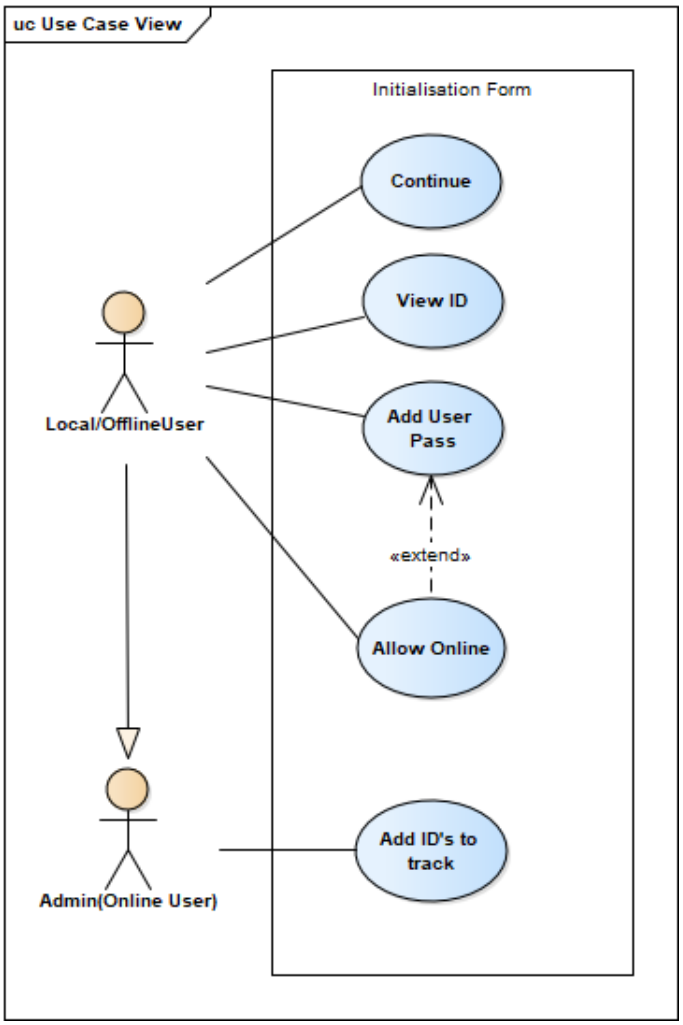

Figure 1.1– Initialization Form.

- My application has 2 types of actors **Local** which works in offline mode as well and **Admin** which is a generalized form of a Local User.
- A Local User can become an Admin after choosing the **Allow Online** option.
- **Allow Online** can be only accessed after inputting a **User Pass**.
- An Admin has the option to add Local Users who he could track data.
- In order to add a Local User the admin has to input their ID and their User Pass.
- Upon initializing the application each Local User is given an ID.
- All types of Users/actors must press continue to go to selection form.

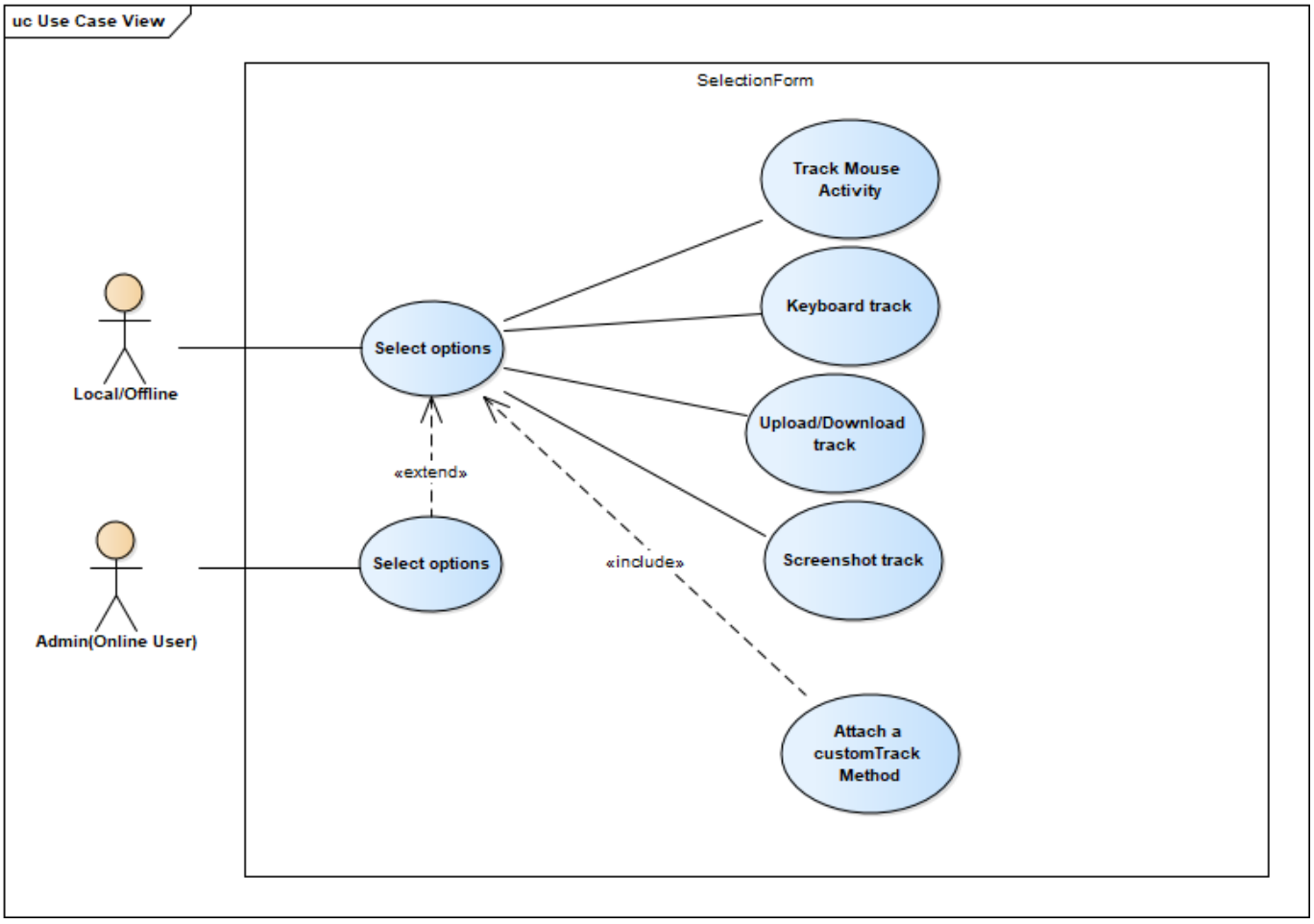

Figure 1.2– Selection Form.

- After completing the initialization form, the Local/Offline User can Select Options from Default Tracking Methods *(Keyboard Track,Track Mouse Activity,Upload/Download Track,Screenshot Track)*.
- Additionally a Local user can add their own Tracking Method, by selecting **Attach a custom Track Method**.
- If a Local user is a part of a network where there is an Admin then the Select Options that he has made will be overwritten by the Select Options that the Admin has on his machine.

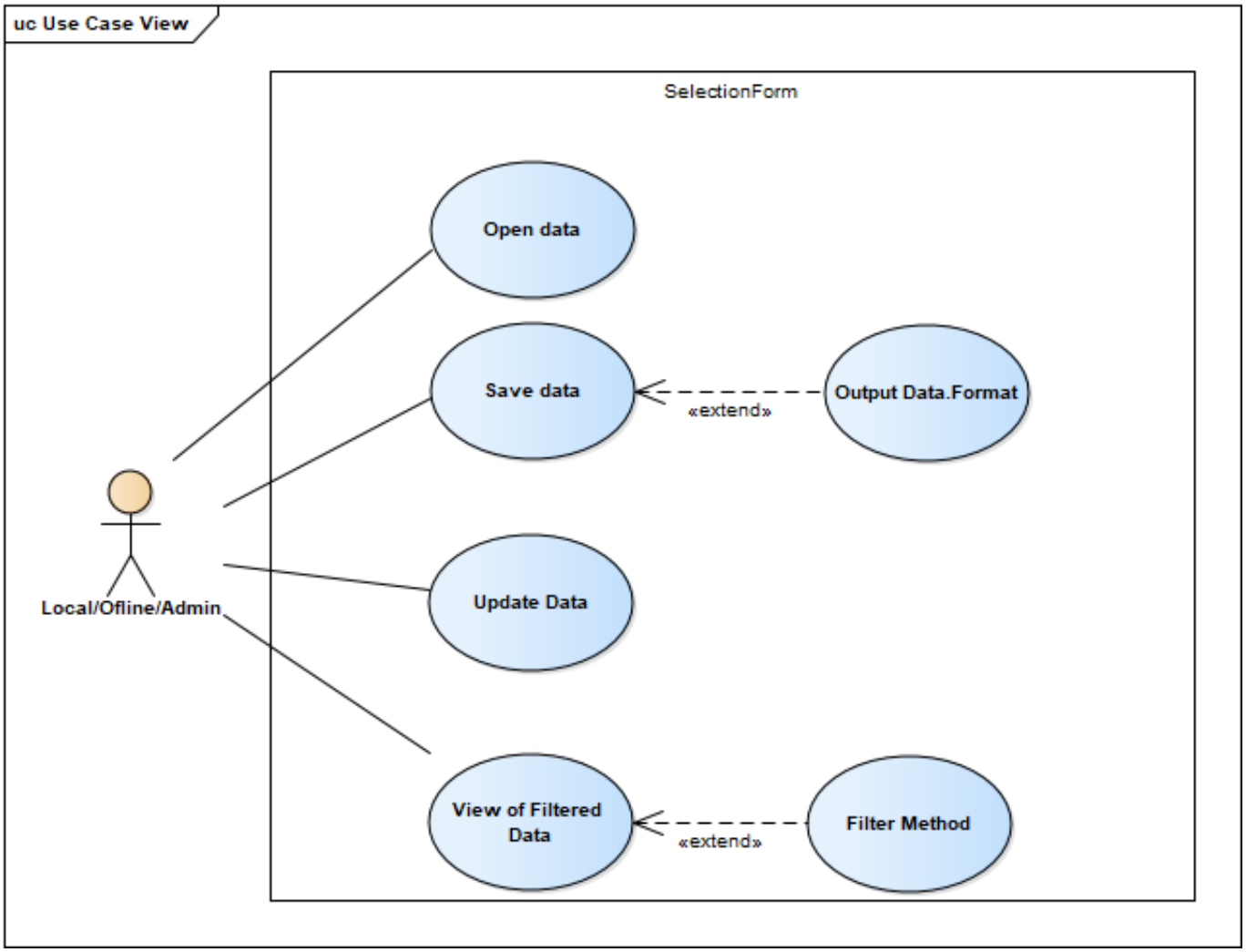

Figure 1.3– Output Form.

After completing the **Selection Form**, the Actor which in this case represents all types of users has 3 options to chose to get the data that was tracked.

- He can View the data that was tracked and update it by pressing Update Data button and apply filtering methods to View the data.
- Or he can save the data that was tracked;If a filtering method was used then data outputted will be influenced by the filter.
- Or he can open and view a data that was saved before.

2 Basic and alternate flows of use cases.

Initialization Form.
Basic flow.(Offline)

- Local/Offline user is given ID.
- Press Continue

Alternate flow.(Local)

- Local/Offline user is given ID.
- Input User Pass and check Allow online
- Press Continue

Alternate flow.(Admin)

- Local/Offline user is given ID.
- Input User Pass and check Allow online
- Add ID and User Pass.(How many Admin desires)
- Press Continue

Selection Form.
Basic flow.(Local/Offline)

- Local/Offline user Selects the tracking methods.

Alternate flow.(Local/Offline)

- Local/Offline user adds a new tracking method.
- Local/Offline user selects the tracking methods.

Alternate flow.(Admin)

- Admin user selects the tracking methods.
- Local/Offline user selections get overwritten.

Output Form.
Basic flow.

- User presses Update.
- User Views the data.
- User Saves the data.

Alternate flow.

- User filters the data.
- User presses Update.
- User views filtered data.
- User Saves the data.

Alternate flow.

- User opens a data file.
- User Views the data.
- User filters the data.
- User presses Update.
- User views filtered data.
- User Saves the data.

3 Model the application using 3 Sequence Diagrams

The following actions in all 3 Sequence Diagrams presented bellow are **common for all users**:

- **AplicationStartup()** represents the initialization of the application and it returns the Interface output to all types of users through **OutputInterface()**
- **Continue(selection)** represents the passing point from initial login form to the selection form.It request the form from the database through **RequestContinue(selection)** and returns the form through **ReturnContinue()**
- **Continue(output)** represents the passing point from selection from to the output form.It request the form from the database through **RequestContinue(output)** and returns the form through **ReturnContinue()** once done the output is show to the user by **OutputForm()**.
- **UpdateView()** from *output form* represents a view update that will give the readable onscreen filtered data to the user by **OutputFilteredView()** it will request the **UpdateView()** from the database apply the filter previously set **"ApplyFilter()"** then a request of data from the TrackMethods will be done "**GetData()** and **GetDataReturn()**" then the data will be returned to the interface through **ReturnUpdateView()**
- **SaveData()** represents a similar execution of **UpdateView()** just that the output will be in a specific data format;It will request the **SaveData()** from the database apply the filter previously set **"ApplyFilter()"** then a request of data from the TrackMethods will be done "**GetData()** and **GetDataReturn()**" then the data will be returned to the interface through **SaveDataReturn()** and the user will get the save the data.*format* through **SaveDataFormat()**
- **OpenData()** is a instance that will be issued by the user to open a data.*format* from his OS and use the **SetFilter()**,**UpdateView()** and **SaveData()** with the context of the data that was opened with **OpenData()** request to the Local OS Database and returned with **OpenDataReturn()** and **OutputDataForm()**

Sequence diagram №1

- **AllowOnline()** is a checkbox for **SetMode(pass,online)** but this option isn't needed nor can be checked for ***OfflineUser*** therefor it is unchecked.
- **MethodsSelection()** represents the selection of tracking messages that is then transmitted to the database to ask for their initialization through **InitializationMethods()** and individual initialization through **StartMethods()**
- **SetFilter()** represents the setting of a filter the the data that will be saved in the database and then used in **UpdateView()**,**OpenData()** and **SaveData()** through **ApplyFilter()**.

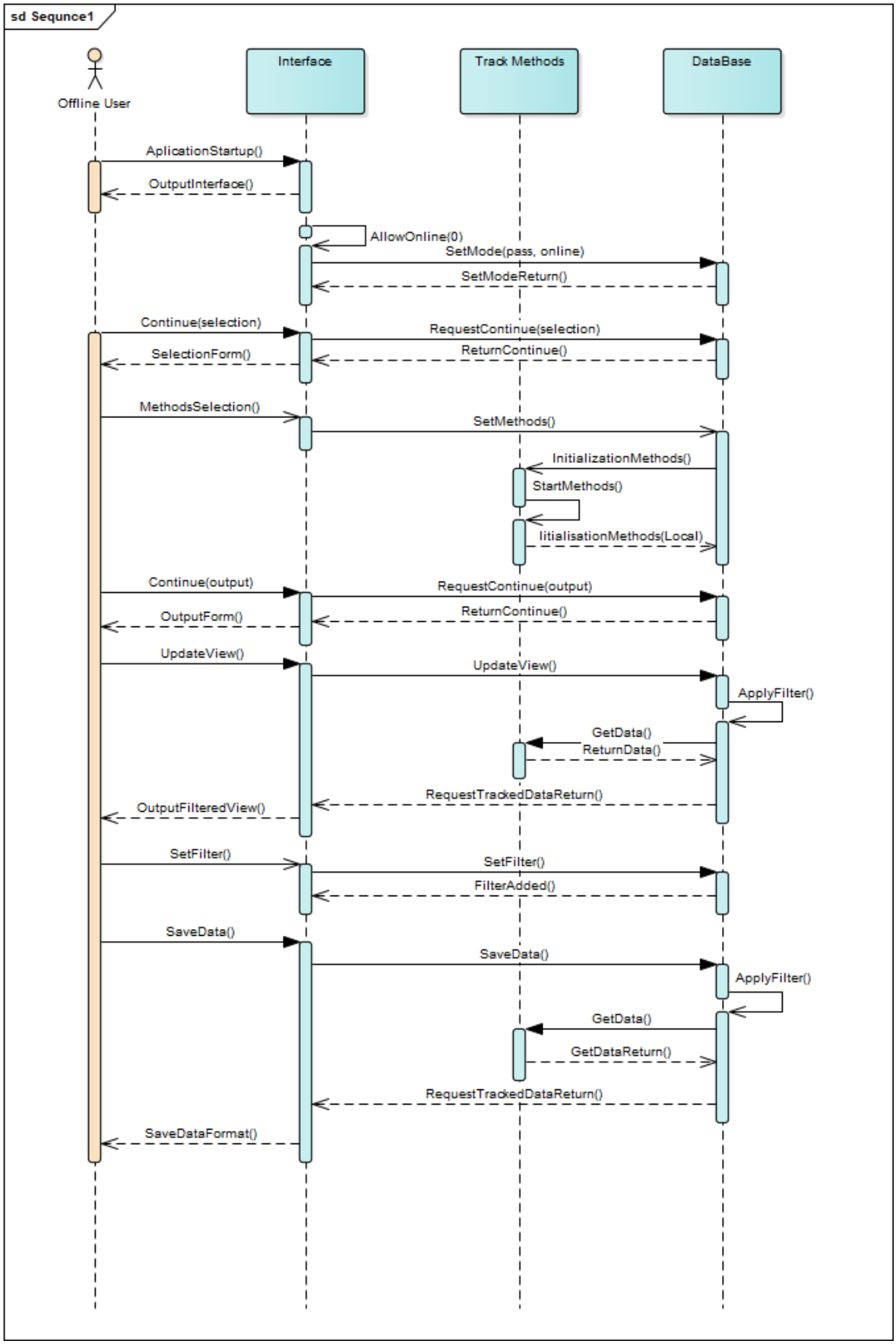

Figure 3.1 – OfflineUser

Sequence diagram №2

- **Addpass()** represents the input of a password in the initialization from where the user sets the mode he wants to work in; As the user is operating in **Online(Admin)**:
  - **SetPass()** sets a security password in the database, which allows checking the **AllowOnline()** to be checked and afterwards saving the mode through **SetMode(pass,online)** in database which will allow to access **AddTrackID(ID,Pass)**.
  - **AddTrackID(ID,Pass)** represents an option to add as many ID's with their respective password to track.
- **MethodsSelection()** represents the selection of tracking messages that is then transmitted to the database to ask for their initialization through **InitializationMethods()** and individual initialization through **StartMethods()** and will return the activation response through **InitializationMethods(Local)** afterwards it will send and request to the local users that the admin has choosed to track **AdminMethods(Send)** and will receive through **AdminMethods(return)** if the user is connected and will see the output on screen through **LocalUserConnected():Yes**.
- **SetFilter()** represents the setting of a filter the the data that will be saved in the database thorough **SetFilter()** function which will afterwards send the filter to the local users and request the filtered data **RequestData()**,**ReturnRequestData()** and then used in **UpdateView()**,**OpenData()** and **SaveData()** through **ApplyFilter()**.

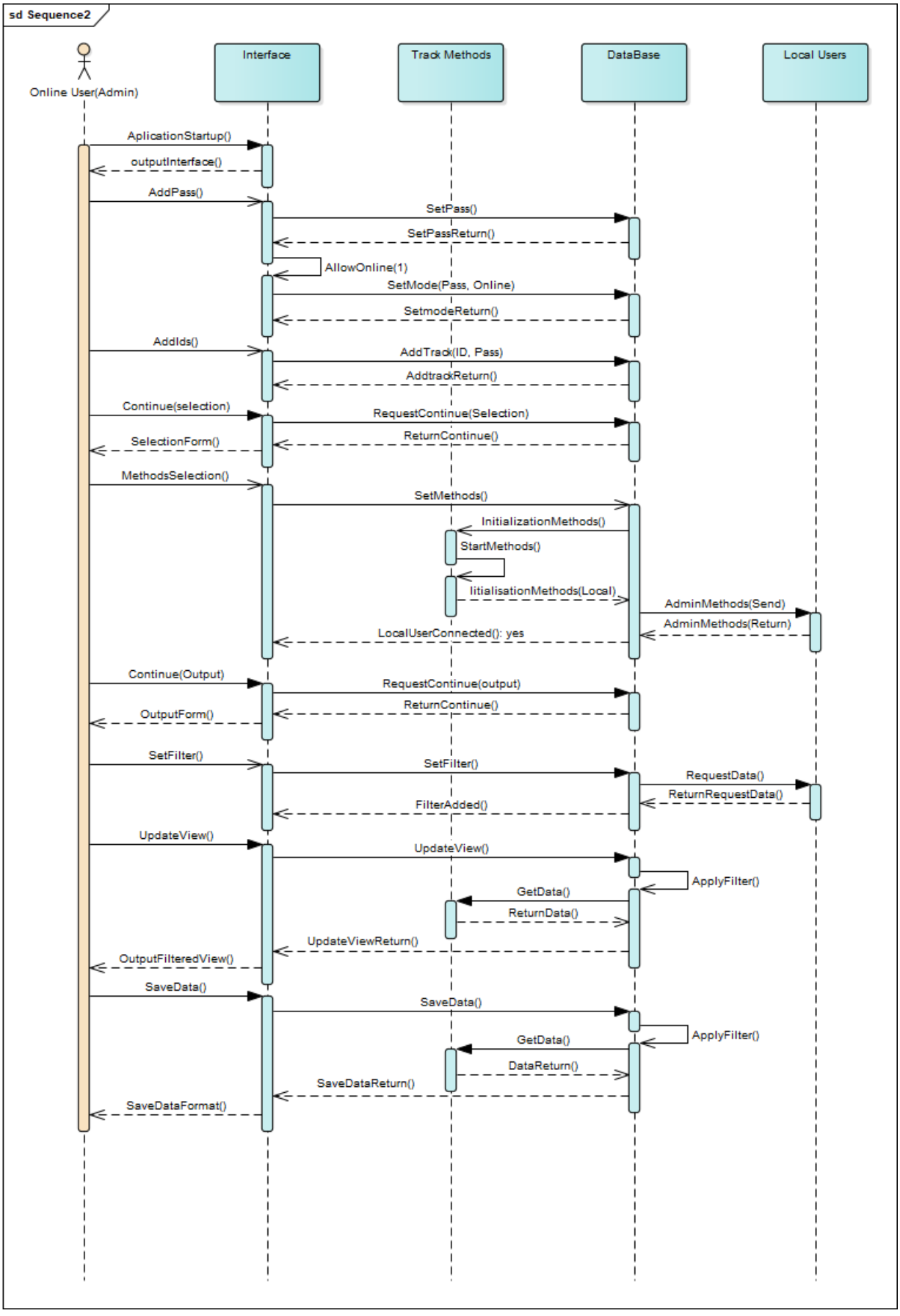

Figure 3.2– OnlineUser(Admin)

Sequence diagram №3

- **SetPass()** sets a security password in the database, which allows checking the **AllowOnline()** to be checked and afterwards saving the mode through **SetMode(pass,online)** in database.
- **MethodsSelection()** represents the selection of tracking messages that is then transmitted to the database to ask for their initialization through **InitializationMethods()** and individual initialization through **StartMethods()** and will return the activation response through **InitializationMethods(Local)** afterwards if it will receive the **AdminMethods(send)** from the admin of the group which will ask for initialization of the methods that weren't initialized through **InitializationMethods()** and individual initialization through **StartMethods()** and will return the activation response through **InitializationMethods(Admin)**
- **SetFilter()** represents the setting of a filter the the data that will be saved in the database thorough **SetFilter()** function which will be used in **UpdateView()**,**OpenData()** and **SaveData()** through **ApplyFilter()**.

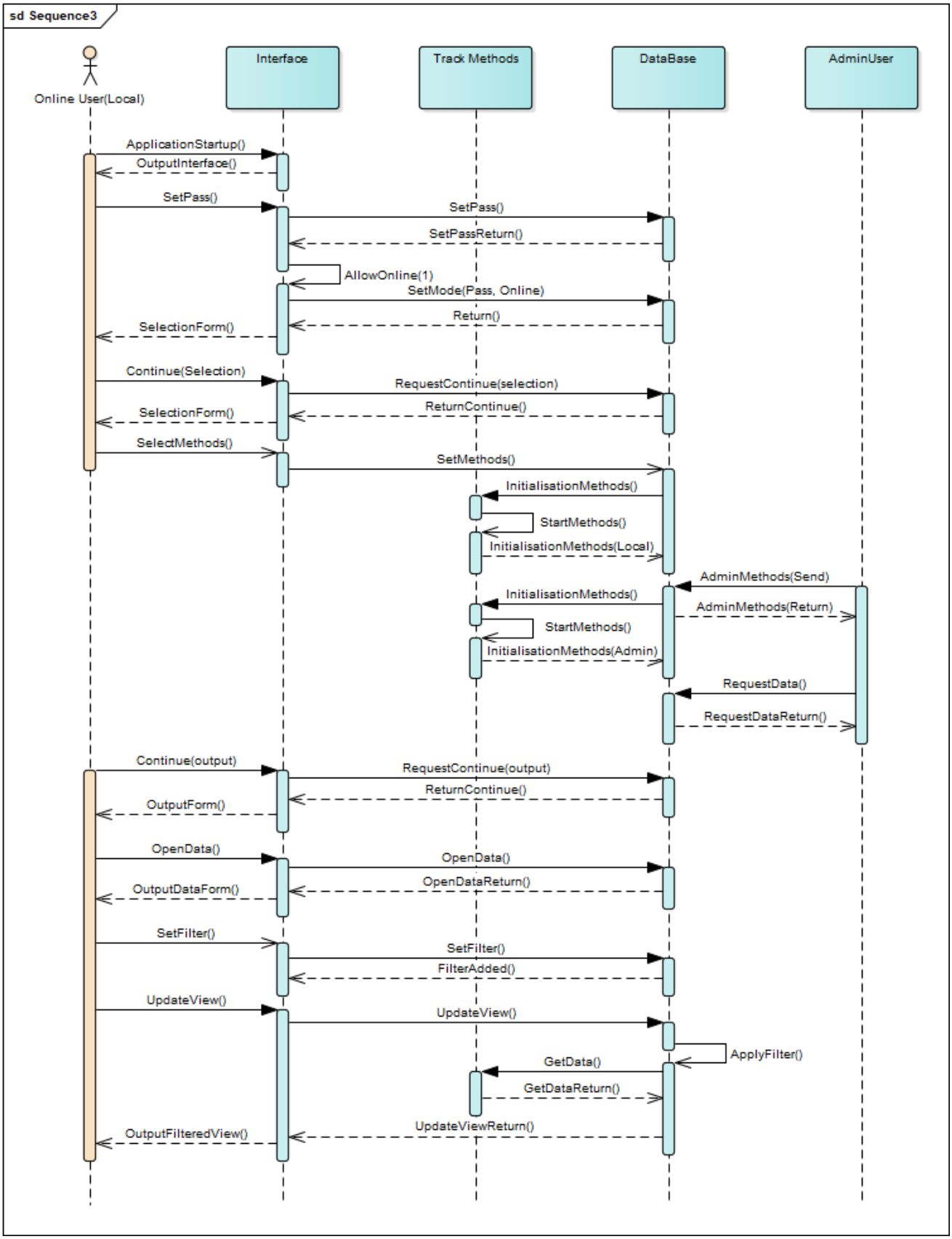

Figure 3.3– OnlineUser(Local)

4 Analyze the Functional and Non-Functional Requirements for the project.

Functional Requirements

- For Track Methods to save tracked data in statistic form.
- For Track Methods to ask for the permissions from the user of what data can be tracked.
- For Filter to show what data can be accessed and requested from the database.
- For Showing the user how to use the Output form.

Non-Functional Requirements

- Product Requirements:
  - Number of Processor: 2
  - Disk capacity min: 6Gb
  - Operating system: Windows ,Ubuntu,Mac Os
  - Database vendor: Microsoft(MySql)
- Efficiency requirements:

  The application should operate with local users tracked data connected with an admin in within a reasonable time.
- Portability requirements:

  The application should run all modern operating systems and be able to interface major relational database systems from various vendors.
- Privacy requirements:

  The application should not reveal private passwords nor any tracked data to outside of network local/admin users.

5 Model of the project with Collaboration Diagrams.

The Figure 5.1 shows communications in between an Admin online user and the interface & database that happen in the Initialization form.

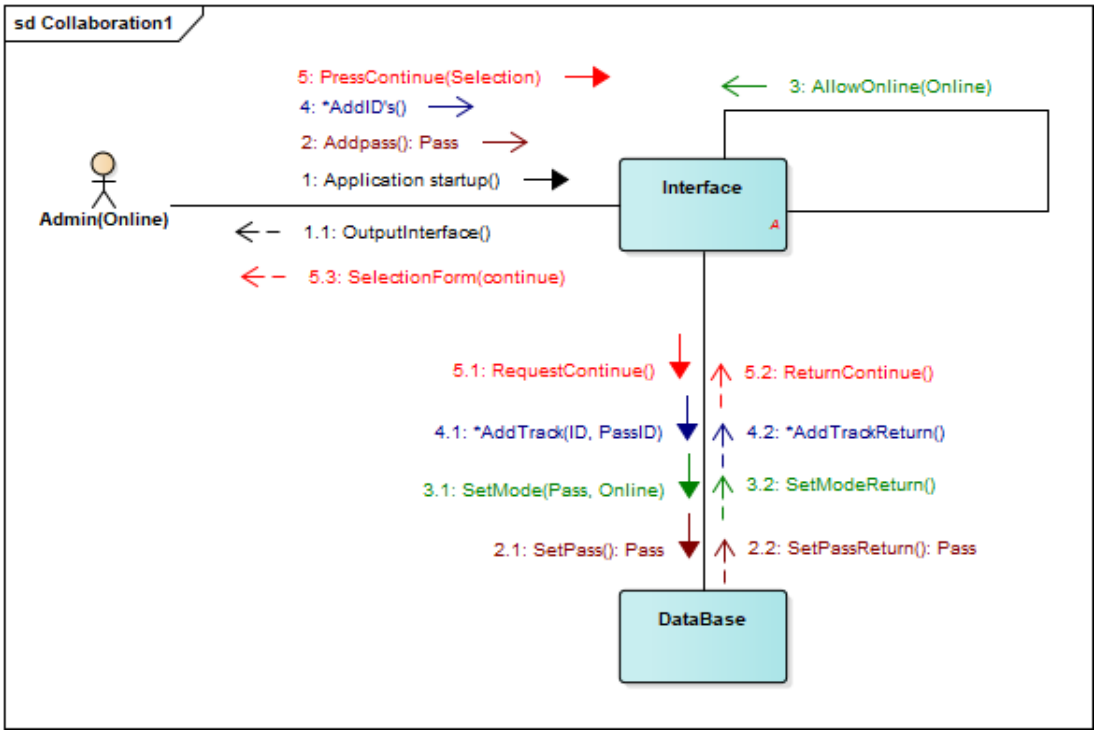

Figure 5.1– Initialization Form

1 User initializes the application.

1.1 The application shows the interface to the user, showing its own ID.

2 User inputs a password in order to go online.

2.1 The interface sends to the database the password.

2.2 Returns the control to the user.

3 User can now go online, by checking the allow online checkbox in the interface.

3.1 The interface sends to the database the online mode initialization

3.2 The user can now proceed to become an admin.

4 User inputs ID's of other users.

4.1 The password and ID inputted in the interface registers the tracked person.

4.2 Returns control to user to input other local users to track.

5 User presses continue to proceed to next form.

5.1 Interface requests the continue method from the database.

5.2 Database returns the continue method therefor continuing to Selection form

5.3 Selection form is outputted to user.

The Figure 5.2 shows communications in Selection form that happen between an Admin online user, the interface that exchanges data with database which dictates the track method and other local user functions.

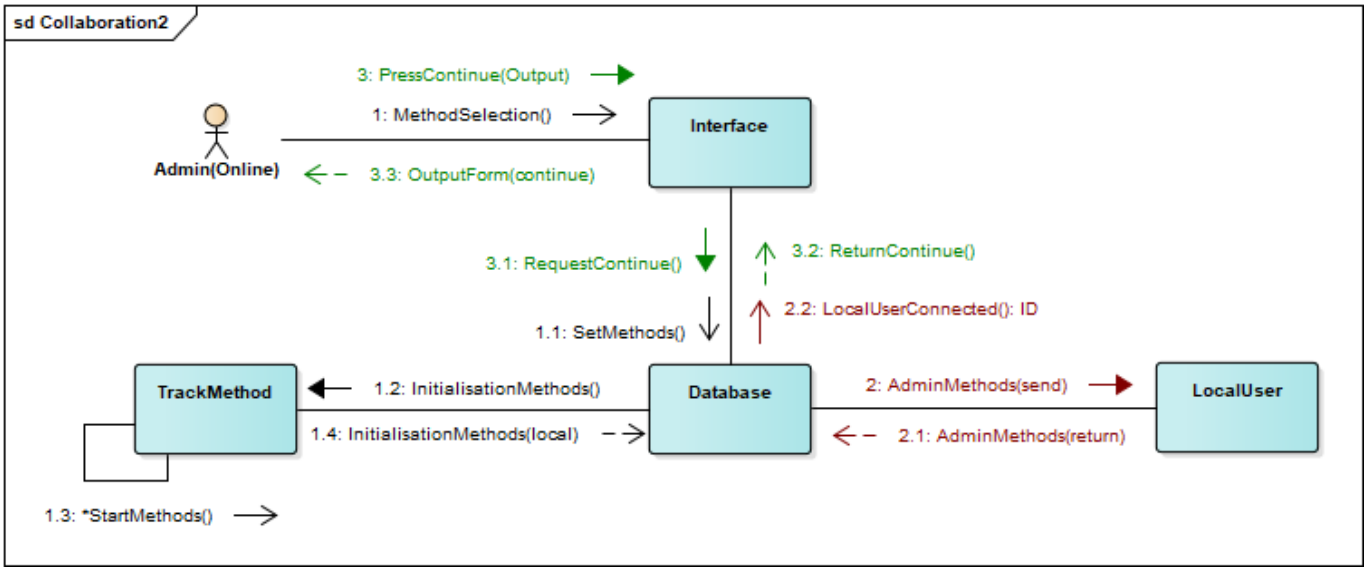

Figure 5.2– Selection Form

1 User selects the track methods he wants to use.

1.1 The interface Sets the methods in the database.

1.2 Database Requests initialization of the track methods

1.3 Each track method is started in the track method to start tracking data from other local users or the pc itself.

1.4 The track method component sends to the database that the local methods were initialized.

2 The database sends the Methods to Local Users.

2.1 The database recives the id of the users that are currently present in the LAN.

2.2 Interface shows which users are currently connected in LAN.

3 User Presses button continue for the output form.

3.1 Interface requests from the database the output form.

3.2 Interface recieves the output form.

3.3 User sees the output form.

The Figure 5.3 shows communications in Output form that happen between the same components as in Figure 1.2.

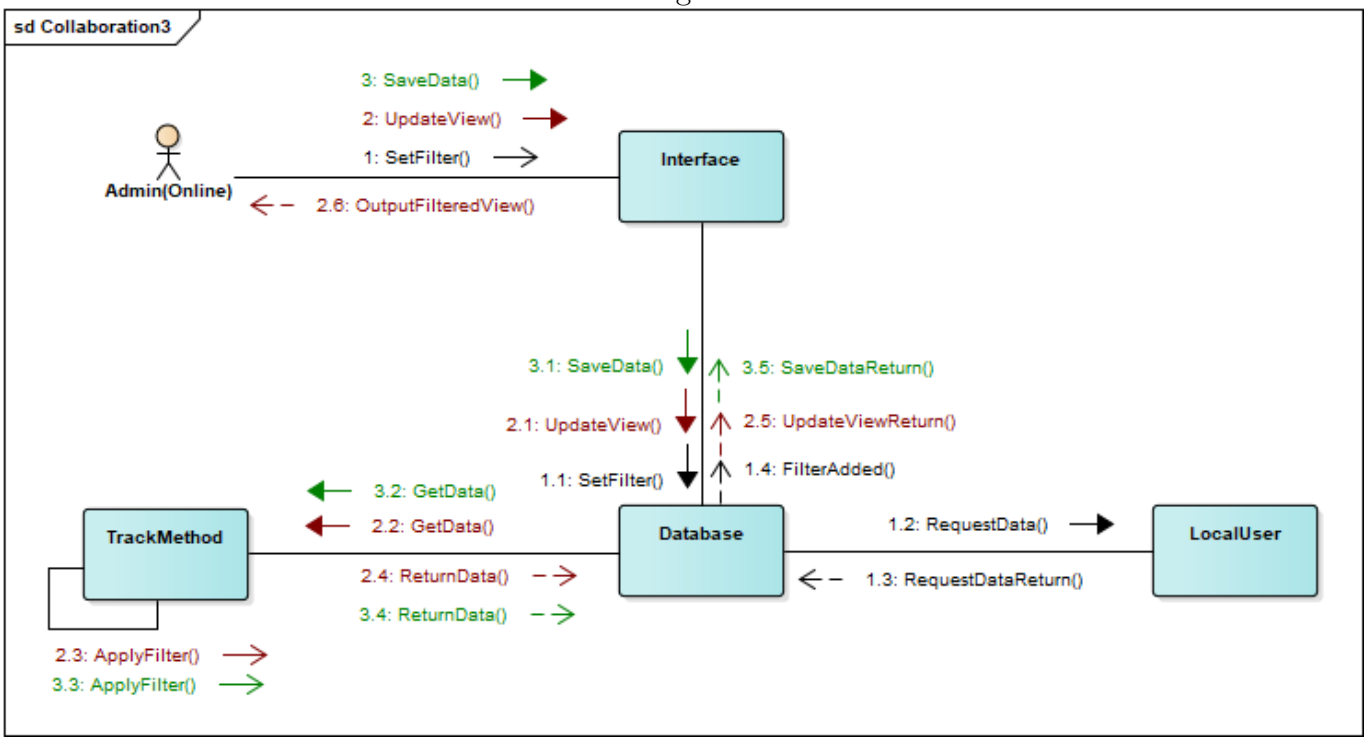

Figure 5.3– Output Form

1 User types a filter for data in the iterface.

1.1 Interface saves the filter into database.

1.2 The database requests filtered data from the local users.

1.3 Local users send the filtered data to the admin Database.

1.4 Interface recieves confirmation that the filter was added.

2 User updates view to see the data collected.

2.1 Interface request for updated view.

2.2 Database requests data from tracking methods.

2.3 Track method component applies filter to the collected data.

2.4 Filtered data is returned to the database.

2.5 Update view outputs the collected filtered data.

2.6 User views the data.

3 User presses button to save of data.

3.1 Interface requests data from the database.

3.2 Database requests data from tracking methods.

3.3 Track method component applies filter to the collected data.

3.4 Filtered data is returned to the database.

3.5 The save file is made outputting a confirmation message on the interface.

6 Technologies planned on using in the project

The technologies planned for using in this project are the following:

- **Operating system:** Windows,Linux,Mac OS,Android,Windows phone.
- **Programming languages:** C,C++, Sql.
- **IDEs:** Qt(Cross-platform framework).

Qt is a cross-platform application framework that is used for developing application software that can be run on various software and hardware platforms with little or no change in the underlying codebase, while still being a native application with native capabilities and speed.

Cross-Platform means that the application should work under all operating system.

As For programming languages C and C++ are the fastest compiling high level languages, and Sql is a system query language which helps with in working with the databases.

7 Model of the project with Class Diagrams.

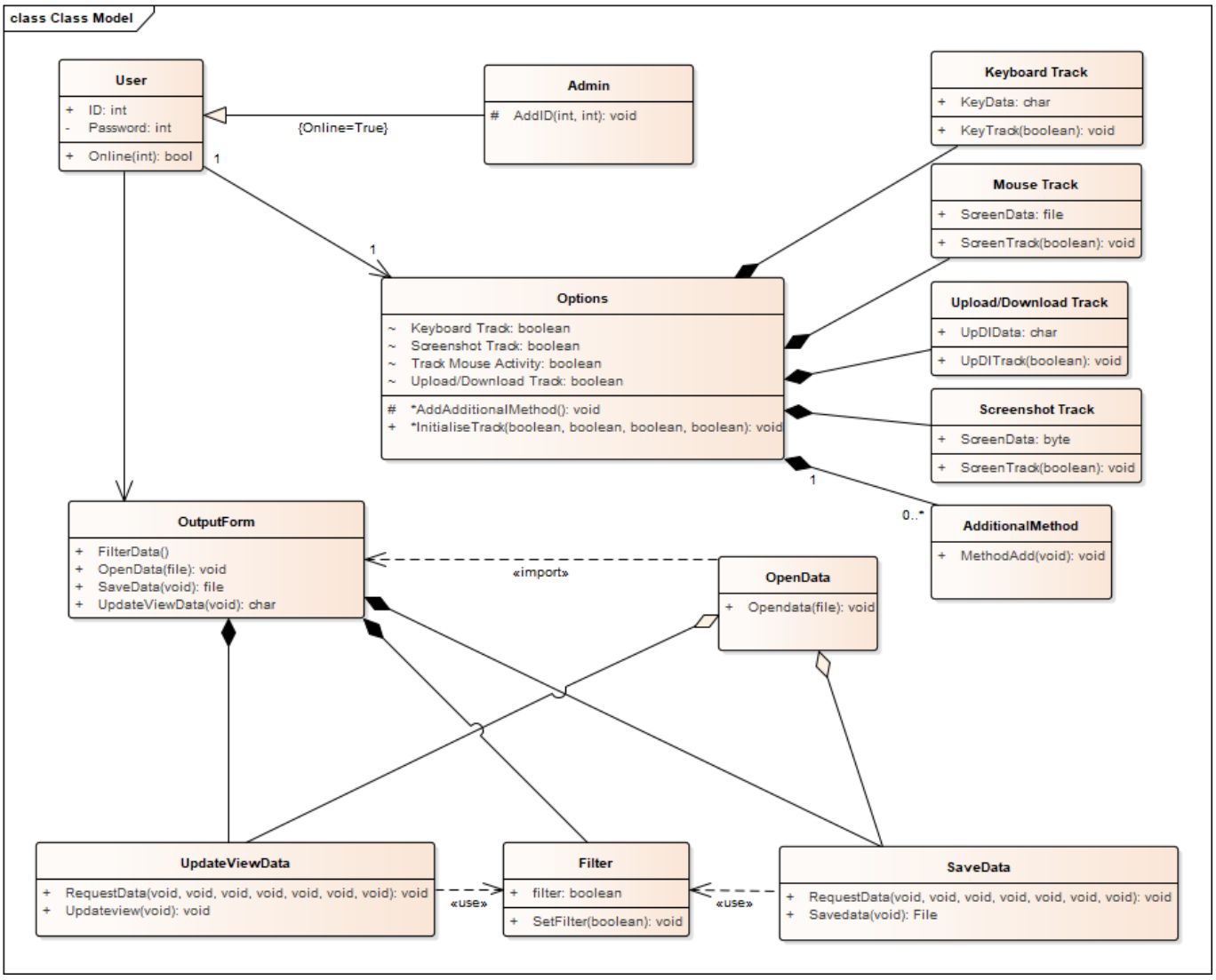

Figure 7.1 – Class Diagram

The class diagram in **Figure 7.1** contains the general conceptual modeling of the application classes used.

The most important classes in this project are:

- User class.*(Class that responds for the user functionality)*
- Options class.*(Class that responds for the option selection and tracking methods instantiation)*
- OutputForm class.*(Class that responds for the tracked data operations)*

The User can be further generalized in Admin class that overwrites the User variables and has the ability to add other Users Id's to track.

Each user is associated with a option selection class and through it the user selects the tracking methods and then after the selection track methods are initialized; if the user choose an additional method then .

The OutputForm is the class that deals with data manipulation and its representation.

– Open data responds for opening the files that contain tracked data and use the other other functions of generalized class OutputForm.

– UpdateViewsData outputs on interface on screen that is processed and requested from the track methods and filter.

– Filter filters the data according to the user input.

– Save data saves the data that is processed and requested from the track methods and filter.

8 SWOT analysis of the project.

The SWOT analysis of the project is explained below in the Figure 8.1 .

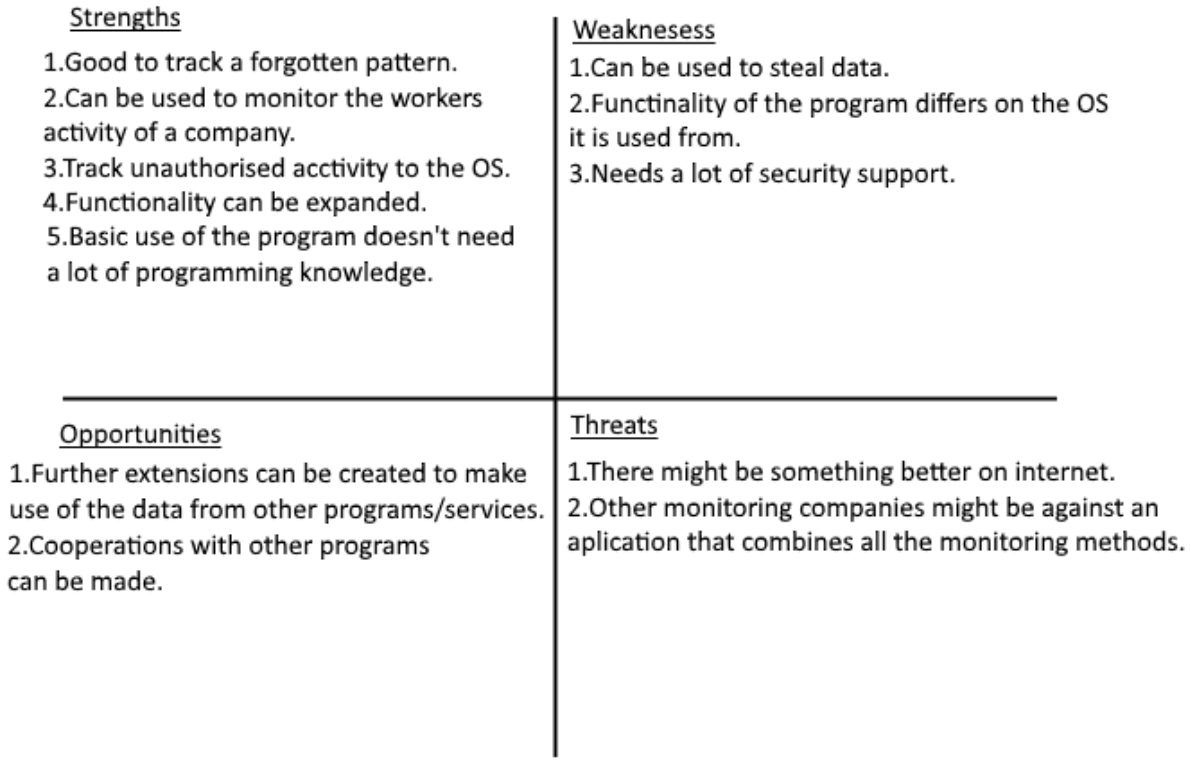

Figure 8.1 – SWOT

9 Model of the application using 3 Statechart Diagrams

In figure 9.1 is presented the state chart which contains the initialization form of my application.

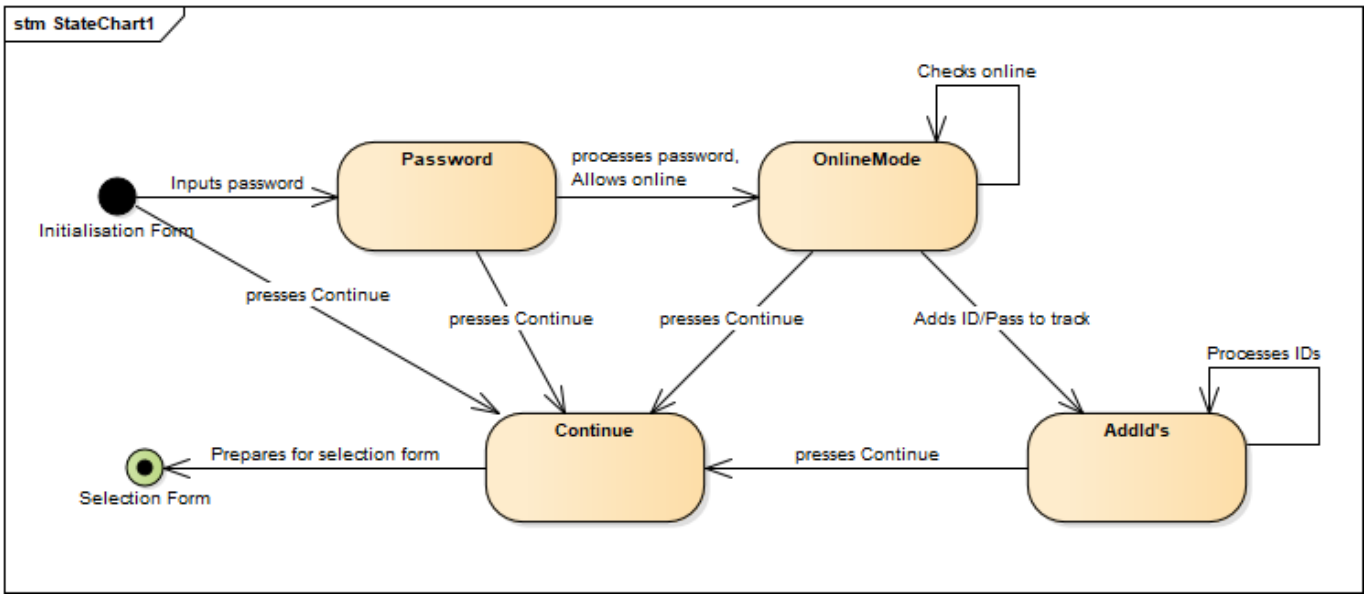

Figure 9.1– Initialization

In figure 9.2 is presented the state chart which contains the selection form of my application.

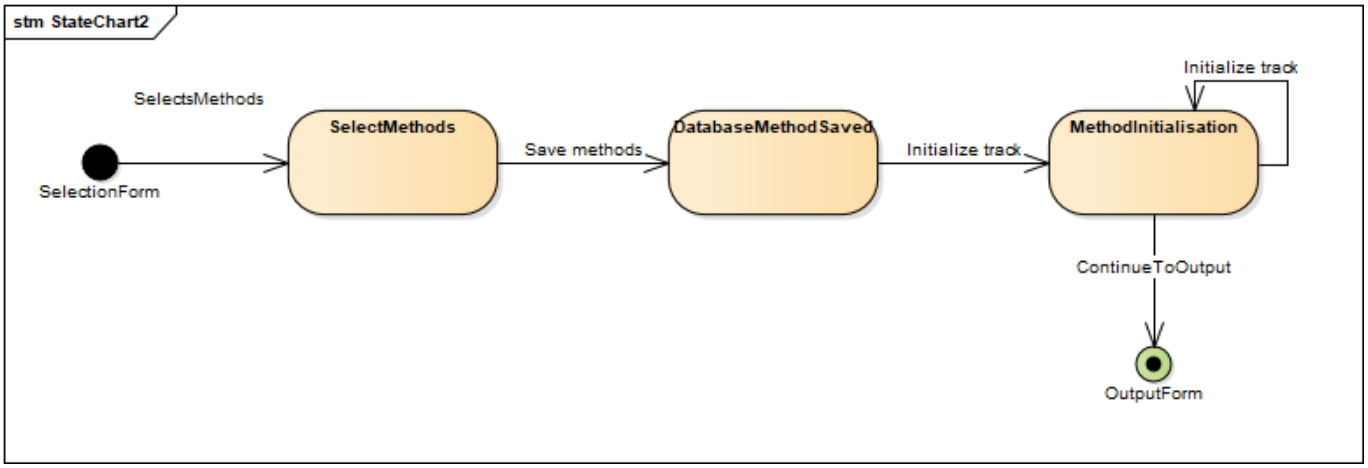

Figure 9.2– Selection

In figure 9.3 is presented the state chart which contains the output form of my application.

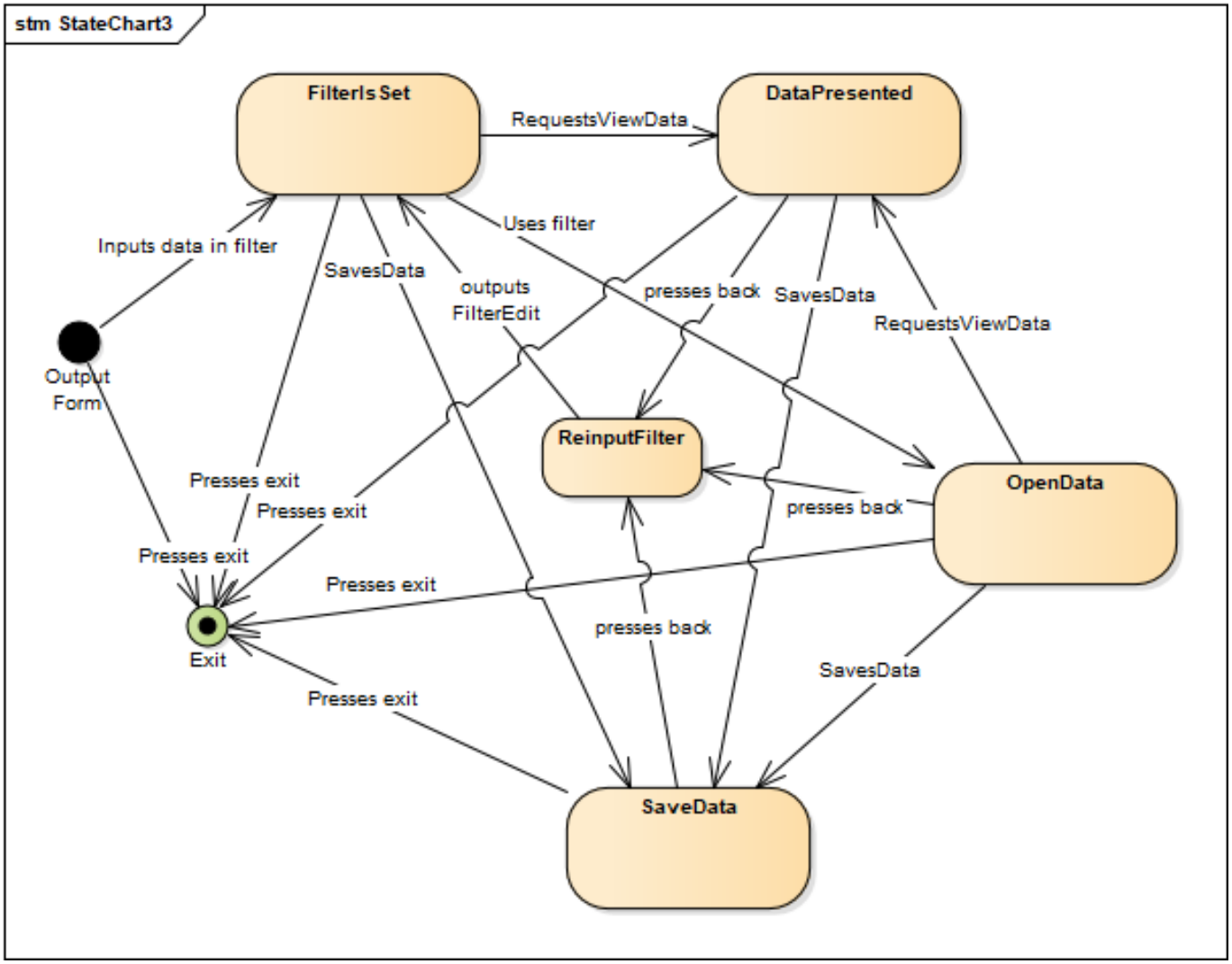

Figure 9.3 – Output

10 Domain analysis of the project

- **Domain description.**

  A tracking system is used for the observing of persons or objects on the move and supplying a timely ordered sequence of location data for further processing.Regardless of the tracking technology, for the most part the end-users just want to locate themselves or wish to find points of interest. The reality is that there is no "one size fits all" solution with locating technology for all conditions and applications, therefor i added the Additional Methods to the selection.

- **Theme importance;**

  The importance of these theme is mainly to make observations on the statistics we are provided with.Because the theme has a big spread and utilization in the read world there the importance of this theme is equal to general need.As far judging from the main functions implemented in my application the importance factors that i thought of are: *to be able to trackback to a specific moment, to prevent of see access of a foreign user in the System and to monitor the how a company is prospering according to how are employees executing their tasks.*

- **Description of existing systems(min 3);**

  - **EXO5** (A business package that's more focused on locking down your data)

    Specs: RemoteKill file encryption, drive lock, curfew, geolocation, logs, data export, RiskSense alerts

    Most of its features are geared more towards an individual or company that wants to make sure that a laptop is being used for the right purposes. Of most use is the incredibly handy RemoteKill option. This enables you to encrypt files and folders remotely if the laptop is stolen. Presets such as 'All Microsoft Outlook.pst files' make it quick and easy to secure important info. You can also add a boot sector lock to shut down the device - and both can easily be reversed if the laptop is recovered.

  - **GlassWire** free firewall software and network monitor can detect threats other miss. Download GlassWire free firewall now to protect your computer.

    GlassWire's network monitor visualizes your current and past network activity by traffic type, application, geographic location, all on a beautiful and easy to understand graph. GlassWire reveals hosts that are known threats, unexpected network system file changes, unusual application changes, ARP spoofing, DNS changes, and alerts you to the problem so you can take action. GlassWire can also remotely monitor and help protect servers or other computers far away.

  - **Norton AntiVirus** (is an anti-malware software developed and distributed by Symantec Corporation since 1991 as part of its Norton family of computer security products. It uses signatures and heuristics to identify viruses. Other features included in it are e-mail spam filtering and phishing protection.)

- **System comparison;**

  As to what my application is capable of comparing it to other applications mentioned above it is capable of implementing the usage of other applications with additional methods and also the download/upload track method is similar to what GlassWire is doing, but as it is a operating system track system it is less focused on tracking the exact locations and data loss prevention unless implemented in the default tracking methods.

- **Scope and objective of the chosen theme.**

  The scope and objective of my theme is to make a system that combines most tracking methods that are already used or adds new tracking methods which aren't yet implemented, and delivers a report easy to understand.

11 Model of the application using 3 Activity Diagrams

In figure 11.1 is presented the state chart which contains the output form of my application.

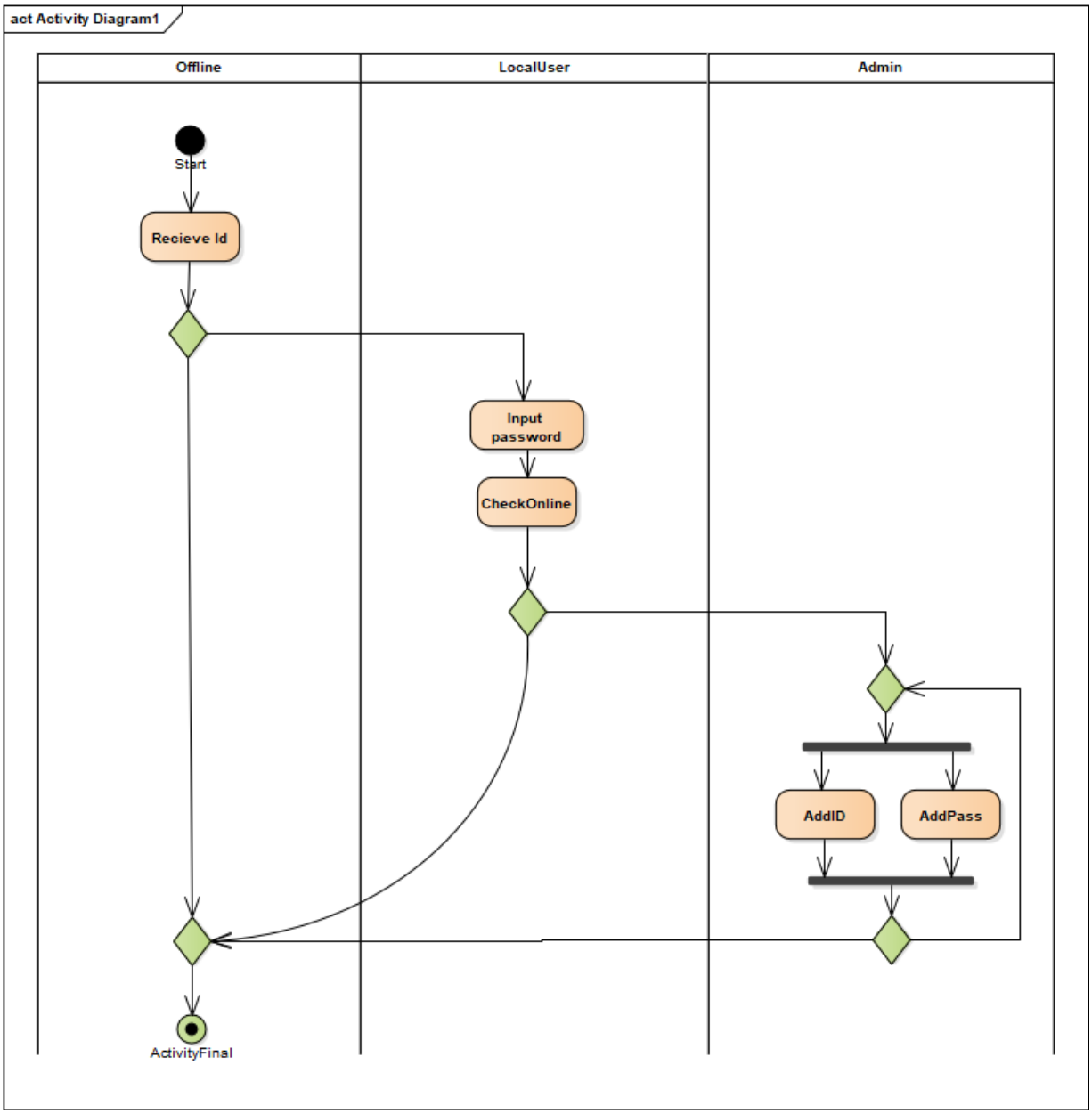

Figure 11.1 – Initialization

Here the user chooses if he wants to work online by adding the password and then checking online,after what he can add Id's to track after what he will track these Id's data from selection form.

In figure 11.2 is presented the state chart which contains the output form of my application.

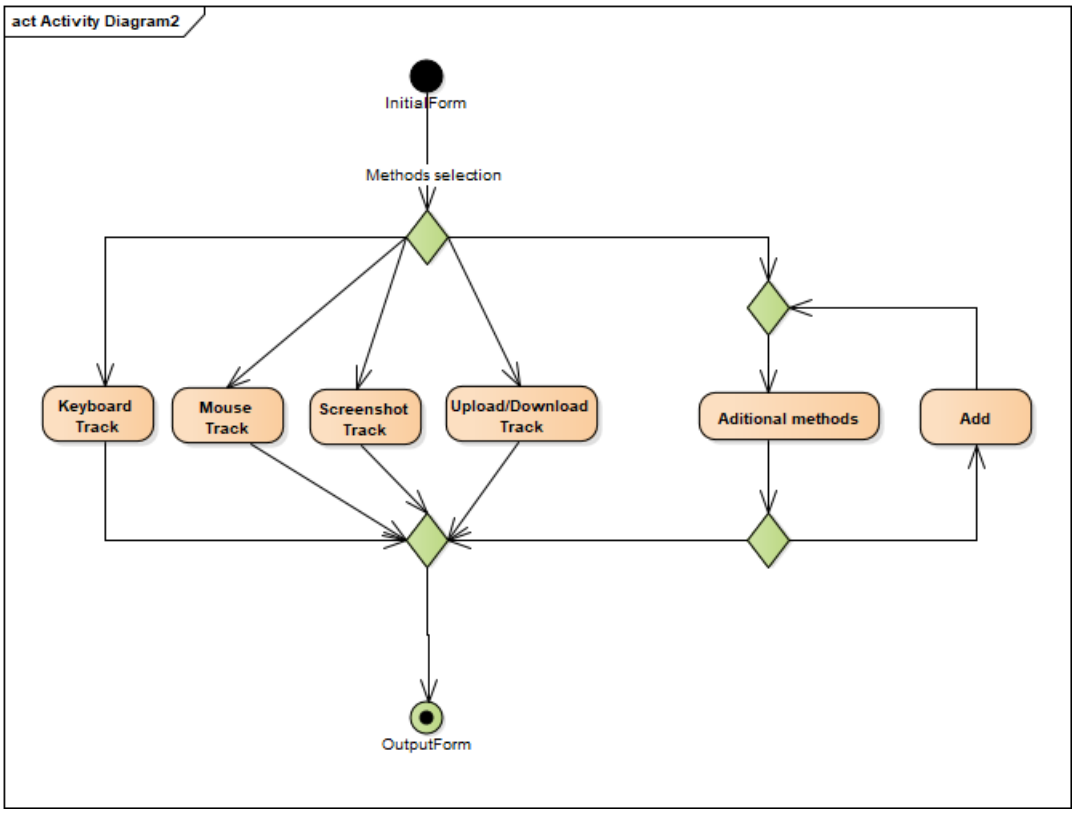

Figure 11.2– Selection

Here user is given methods to select and and continue to work with in the output form.

In figure 11.3 is presented the state chart which contains the output form of my application.

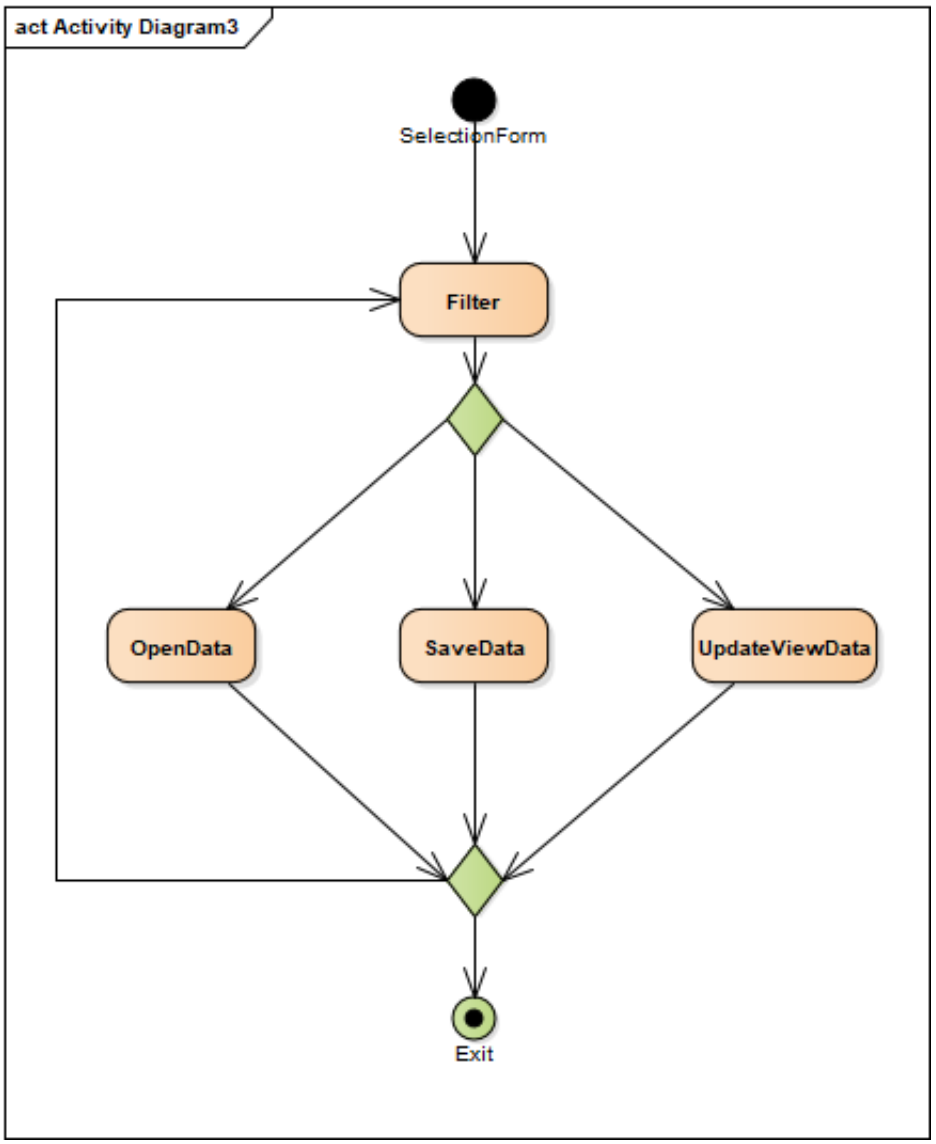

Figure 11.3– Output

User can choose how to save his data collected from the tacking methods.

12 Activity Map of the application.

The Activity Map that my application has is based around:

- Initialization(Where the user selects the mode he wants to operate in)
- Selection(Where user selects the Tracking methods he wants to use)
- Output(Where user selects how he wants to output the data that was tracked)

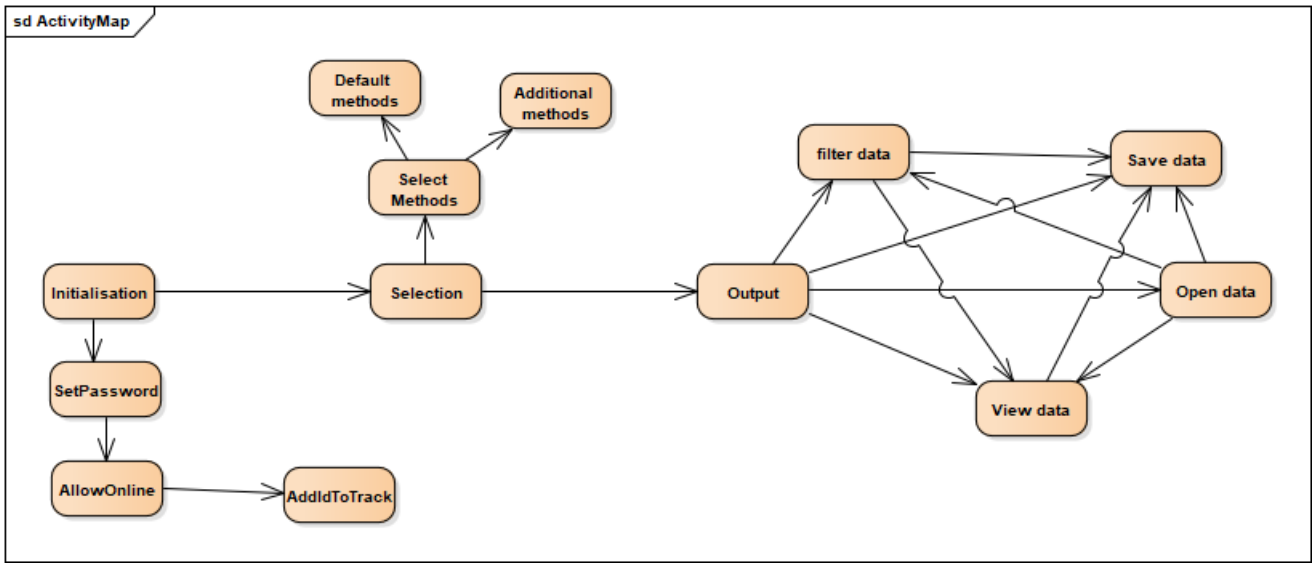

Figure 12.1– Output

13 Model of the application using Component Diagrams

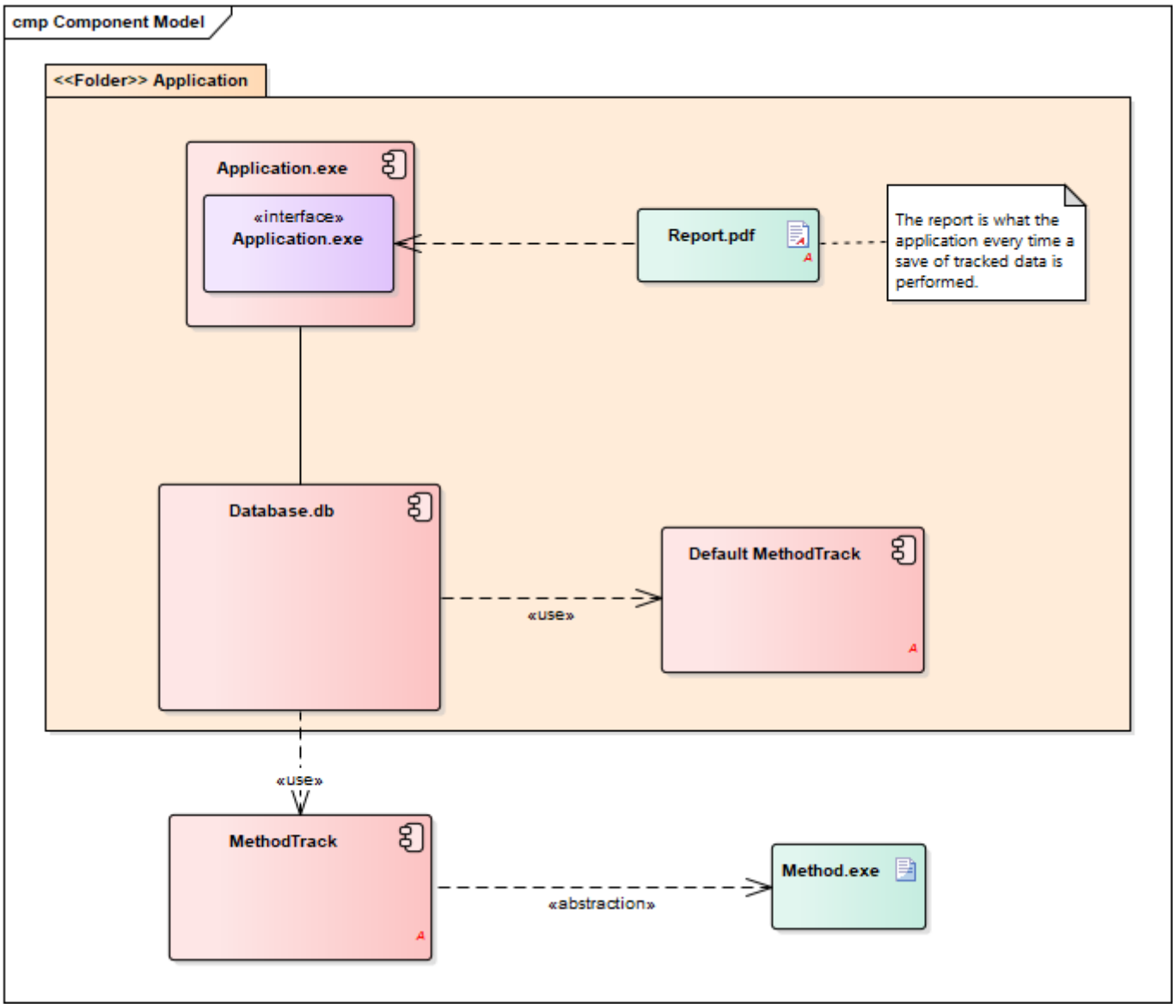

Figure 13.1 – Class Diagram

The application executable in fig 13.1 is contained in its own specific folder and by running it we get the interface. The database can use methods from outside the application folder and the default ones. Each time a save is performed the the application generates a Report.pdf that outputs the tracked data in an easy way for the user to understand.

14 Steps needed to setup the project.

In order to setup the project:

- Download and install Qt(framework)
- Download and install Sql management studio.
- A team that is specialized in operating with data under windows,Mac os and Linux to create the default methods that the application is sought to provide by default.

15 Model of the application using Deployment Diagrams

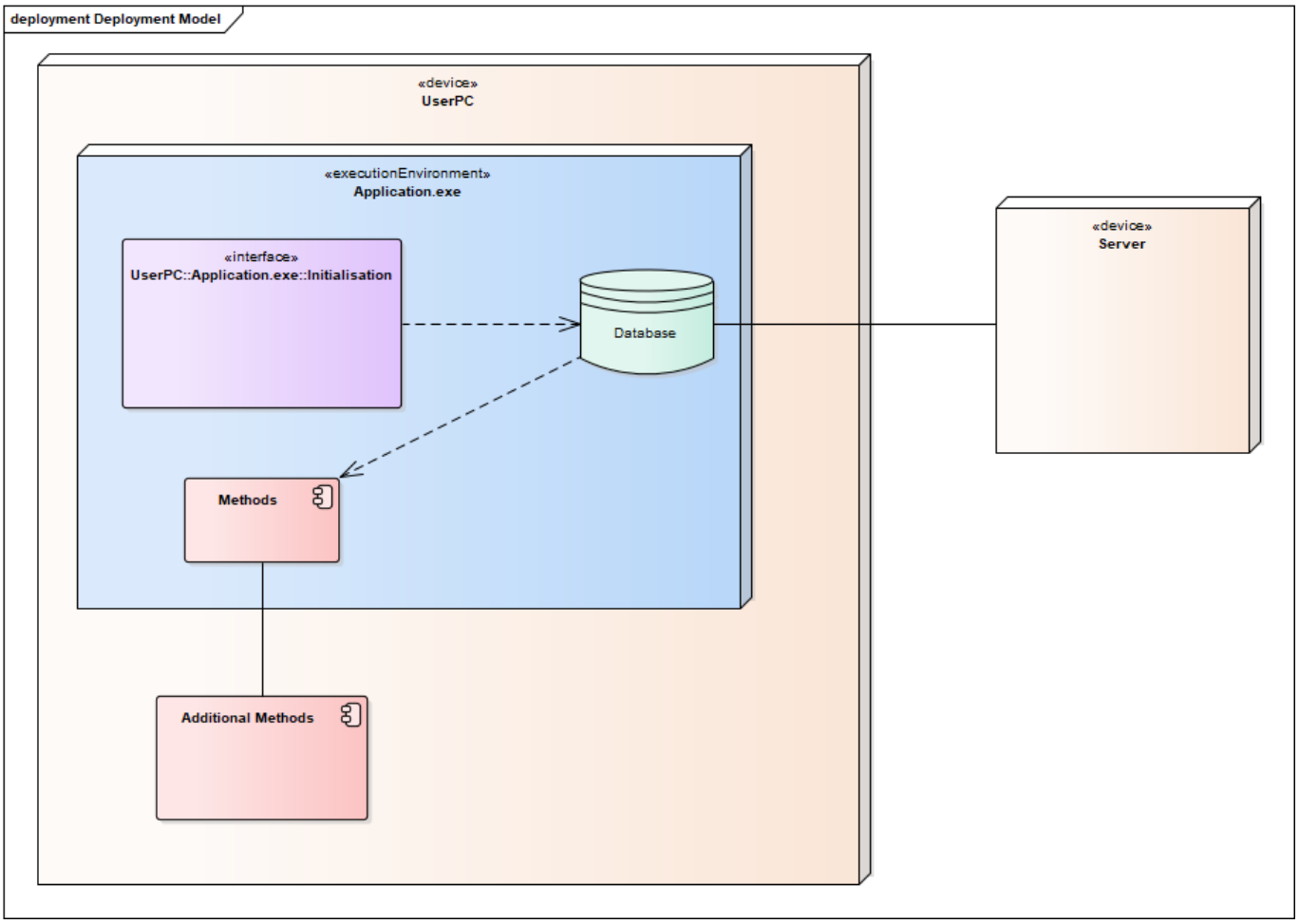

Figure 15.1 – Class Diagram

The application in Figure 15.1 is run on the device it was installed ,it communicates with the server through database if it was connected to LAN ,the database also communicates with the Methods and tells them when to start their process.

16 Document the application delivery / installation.

The application is delivered through and online shop.

The user is provided with a install package executable, the installer will unpack the executable, the package with the methods and a file containing the database(that will register the commands of user through the interface and the track methods tracked data).

The installer will ask the user the installation directory ,ask if he wants to place a shortcut of the executable on the desktop and upon the ending of the installation will be asked if he wants to run the program.

Conclusions

In this course work i learned how to setup and application project ,how to describe it using Unified Modeling Language ,what an application project needs in order to be understood by the team members and how to manage the project in order to implement the idea behind it.In my course work i provided a standard design of my Application but as far as i don't start building it in real life i can't provide all the small details about it,therefor this project just interprets the base idea of how my application should work look and how it communicates with devices.

References

1 Learn Unified Model Language,https://www.tutorialspoint.com/uml/

2 Wikipedia, https://en.wikipedia.org/wiki/Unified_Modeling_Language

3 Notes on UML course given by professor and laboratory assistant.

4 Use Case Examples – Effective Samples And Tips http://www.gatherspace.com/static/use_case_example.html.

5 Sequence Diagrams UML https://www.smartdraw.com/sequence-diagram/

6 Collaboration Diagrams UML https://www.tutorialspoint.com/uml/uml_interaction_diagram.htm

7 UML - Class Diagrams, https://www.tutorialspoint.com/uml/uml_class_diagram.htm

8 UML - Statechart Diagrams, https://www.tutorialspoint.com/uml/uml_statechart_diagram.htm

9 UML - Activity Diagrams, https://www.tutorialspoint.com/uml/uml_activity_diagram.htm

10 UML - Component Diagrams, https://www.tutorialspoint.com/uml/uml_component_diagram.htm

11 UML deployment diagrams, https://www.uml-diagrams.org/deployment-diagrams-overview.html