Garcon

Software Requirements

Specification

Ramazan Selim Şahin 2171999

Gürkan Kısaoğlu 2171726

# Table of Contents

**1.Introduction5**

1.1 Purpose of the System5

1.2 Scope5

1.3 System Overview6

1.3.1 System Perspective6

1.3.2 System Functions9

1.3.3 User Characteristics10

1.3.4 Limitations10

1.4 Definitions11

**2. References11**

**3. Specific Requirements12**

3.1 External Interfaces12

3.2 Functions13

3.3 Usability Requirements27

3.4 Performance Requirements27

3.5 Logical Database Requirements28

3.6 Design Constraints29

3.7 Software System Attributes29

3.8 Supporting Information30

**4. Verification30**

**3. Appendices30**

3.7 Assumptions and Dependencies30

3.8 Acronyms and Abbreviations30

# Table of Figures

Figure 1: Context Diagram.....................................................................................6

Figure 2: External Interfaces………………………………………………………….12

Figure 3: Use Case Diagram................................................................................13

Figure 4: Issue About Cleaning Services Sequence Diagram..............................16

Figure 5: Get Transportation Information Sequence Diagram..............................19

Figure 6: Logical Database Requirements Class Diagram...................................28

# Table of Tables

Table 1\_1: System Functions.…………................................................................9

Table 1\_2: Table of Definitions….…………………………………..……………….11

Table 1: Open Security Issue Function................................................................14

Table 2: Open Cleaning Issue Function...............................................................15

Table 3: Order Food Function……………………………….………………………..17

Table 4: Get Transportation Info Function............................................................18

Table 5: Get Event Information Function…………………….………………………20

Table 6: Close Issue Function…...........................................................................21

Table 7: Register Issue Function..........................................................................22

Table 8: Get Issue Information Function...............................................................23

Table 9: Authentication…………………………………………….…………………..24

Table 10: Add User Function..................................................................................25

Table 11: Block User Function................................................................................26

# Introduction

This document is Software Requirement Specification for Microsoft’s smart campus project, called Garcon.

## Purpose of the System

The purpose of the project is making METU campus smarter than ever. Making the campus an interactive environment for both students and workers. When there is a campus wide security issue, or environment issue students can immediately inform workers of campus with this system. In addition, it enables students to get information about campus transportation and food possibilities.

## Scope

The scope of this project is providing users to talk interactively with campus workers or gathering information about campus. To accomplish this task an embedded system will be developed. Two potential groups of users exists:

● Students that wants to gather information about campus or open a ticket about a security or environment issue.

● Workers which is a group of campus employees that can see and close issues opened by students.

Garcon makes campus an interactive and informative environment with functionalities like gathering transportation data and event data around campus, opening security or cleaning issues and ordering food.

Therefore, the software has four main

products:

● Mail Service

● Server

● Speech to Text Service

● Third party software called Yemeksepeti API

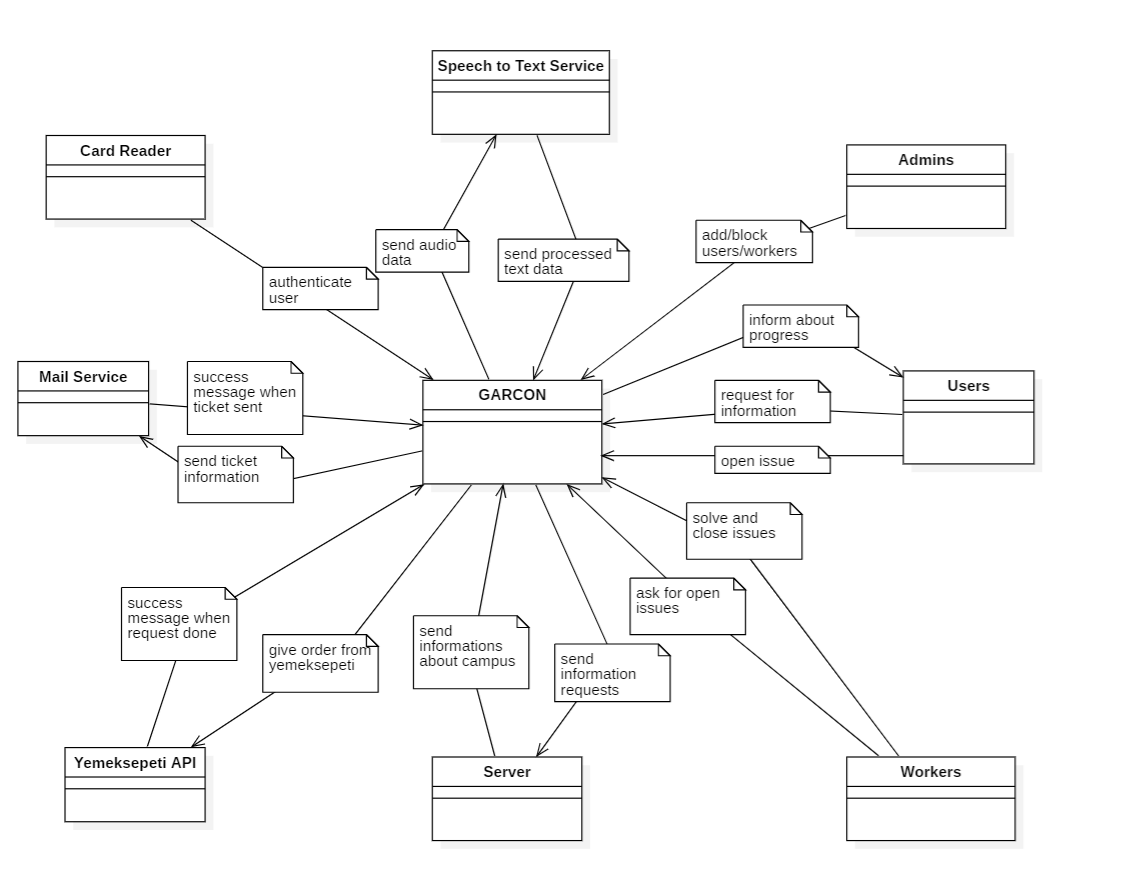
By this software, users will be able to talk with this system to gather information instead of searching on the web or calling some people.

## 1.3 System Overview

This section will give general information about the system.

### 1.3.1 System Perspective

General purpose of this system is making students life in campus much easier than before. System uses a card reader device for student id cards to authenticate students. Then, waits for the student to talk. When student talks, speech to text service analyses the speech and decides whether student opening an issue or asking for an information. After this stage, Garcon will do whatever students want automatically. If an issue opened , mailing service activated; or if an information asked, then Garcon will get the information to the student from various services.

 **Figure 1: Context Diagram**

#### System Interfaces

● **Speech to Text Service:** Garcon system uses speech to text service to analyze what user says, if it is about an issue, or is a request about an information.

● **Yemeksepeti API:** System creates a connection between user and yemeksepeti with this API. Users can easily order food from yemeksepeti without visiting its webpage.

● **Mail service:** When an issue submitted by user, this service activated and service post mail notification to workers about opening issue.

#### User Interfaces

No visual interface available, all user interactions are done via audial inputs and outputs.

#### Software Interfaces

● **DBMS:** Garcon uses a database system to store users and issues that opened by the users.

● **Server:** System uses the server to gather data from external services.

#### Hardware Interfaces

● Device should include a card reader device, microphone device and speaker as hardware to operate normally.

#### Communication Interfaces

Garcon uses HTTP/HTTPS for providing communication between server-device and device-device.

#### Memory Constraints

This is not an important issue for Garcon. System should have enough memory to process audial data and communicate with server.

#### Operations

**User Functions:**

**-**Post security ticket

**-**Post cleaning service ticket

**-**Order food

**-**Get transportation information

**-**Get campus event informations

**Admin Functions:**

**-**Add user

**-**Block user

**Worker Functions:**

-Register to an issue

-Close an issue

-Get information about issues

### 1.3.2 System Functions

|  |  |
| --- | --- |
| **Functionality** | **Description** |
| Post security ticket | A user starts conversation with Garcon about a security issue and Garcon creates an issue. |
| Post cleaning service ticket | A user starts conversation with Garcon about a cleaning issue and Garcon creates an issue. |
| Order food | User wants Garcon to order food from yemeksepeti and Garcon uses external api for this functionality. |
| Get transportation information | User asks for available transportation information and Garcon communicates with server and shows user the information. |
| Get campus event informations | User asks for campus events information and Garcon communicates with server and shows user the information. |
| Close an issue | Worker closes the issue when the issue is handled. |
| Register to an issue | Worker registers an issue and there is a warning system for other workers to prevent register unnecessarily many workers for one issue. |
| Get information about issues | Worker asks for open issues and Garcon communicates the server and shows the data. |
| Add User | Admin adds user to DB. |
| Block User | Admin blocks user from DB. |

Table 1\_1: System Functions

### 1.3.3 User Characteristics

The target users of Garcon system can be categorized into three types as users(students), admins and workers.

Users just need their Campus ID Card to use this system since they will just talk to the Garcon and Garcon will take care of all functionality.

Like users, admins also need an ID Card for adding a user to system or blocking a user from the system.

Workers are expected to interact with system with their ID Card too. They are the employees which closes campus issues or get information about campus issues(like seeing active issues or closed issues).

### 1.3.4 Limitations

● Regulatory policies: Since the Card Reader scans the Campus ID Card, and gets personal information about users; any of the data should not be published to the community.

● Hardware limitations: Since the system operates on an embedded environment, user needs many devices capable of communicating with user and server in campus.

● Interfaces to other applications: Garcon system should be compatible with the other services and APIs it uses.

● Parallel operations: System must be capable of serving multiple users parallel. So, parallelization is a must.

● Audit functions: System does not involve banking hence there is no audit functions.

● Control functions: Controlling database functions(like adding or blocking users) are only available to system admins. Standard users cannot use any control operation.

● High-order language requirements: Java is a good choice of higher order language since it is a multiplatform language with OOP support.

● Signal handshake protocols: System uses HTTP protocol.

● Quality requirements: It is very crucial to keep the data (users’ information) safe; therefore, system should backup regularly.

● Criticality of the application: System failures are very important and system has to be reliable all the time since users have to interact with the device all the time.

● Safety and security considerations: Admins are responsible of the safety of the system by keeping user data secret.

● Physical/mental considerations: There is no physical/mental considerations.

## 1.4 Definitions

|  |  |
| --- | --- |
| **Term** | **Description** |
| User | Students/Academic personnel that uses Garcon. |
| Worker | Campus personnel that uses Garcon for closing issues. |
| Admin | Personnel that can add students/academic personnel or workers to the system. |
| API | Application programming interface. |
| Speech to text Service | A service that can analyse the speech and decide whether it is about an issue or is an information request. |

Table 1\_2: Table of Definitions

# References

**This document is written with respect to the specifications of the document below:**

29148-2011 - ISO/IEC/IEEE International Standard - Systems and software engineering -- Life cycle processes --Requirements engineering.

# Specific Requirements

## 3.1 External Interfaces

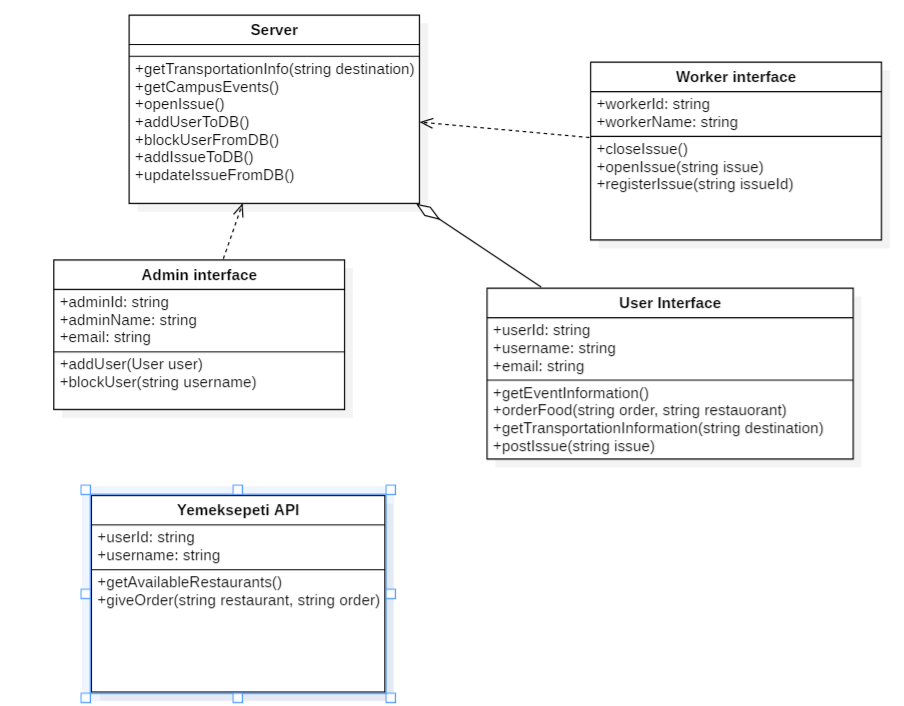


Figure 2: External Interfaces

● **Yemeksepeti API:** Users order food or get available restaurant informations from this API.

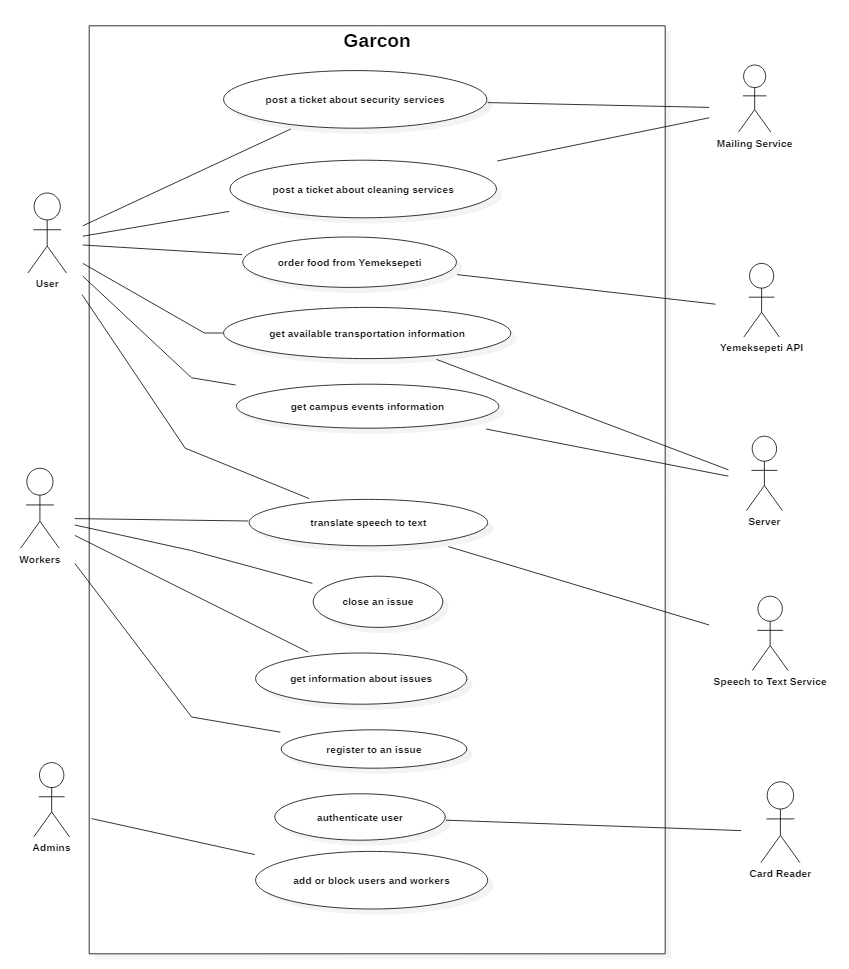
● **Worker’s Interface:** Workers interface can close issue, register an unregistered issue.

● **Admin Interface:** This interface’s main purpose is to make necessary changes in DB related to the students(users).

● **Server:** Server is the part of the program that communicates with DB and make related changes.

● **User Interface:** This interface gives users functionalities like giving order or getting information with help of server.

## 3.2 Functions



**Figure 3: Use Case Diagram**

|  |  |
| --- | --- |
| **Use case name** | Issue about security service |
| **Actors** | Users, Speech to Text Service, Mailing Service, Server |
| **Description** | If a user notices a security issue he/she can notify related workers via Garcon. |
| **Data** | Audial input from user |
| **Preconditions** | User should be authenticated |
| **Stimulus** | User giving audial input about issuing a security request |
| **Basic Flow** | Step 1 – User gives audial input  Step 2 – Audial input gets processed by Speech to Text Service  Step 3 – Issue is created and server side is informed  Step 4 – Processed text of audial input is sent as an email to the related workers  Step 5 – User is informed that issue is registered |
| **Alternative Flow** | Step 3 – System detects same request already issued  Step 4 – Importance level of request is updated  Step 5 – User is informed that issue is already registered |
| **Exception Flow** | - |
| **Post conditions** | An issue instance is created on system and related workers are informed. |

Table 1: Open security issue function

|  |  |
| --- | --- |
| **Use case name** | Issue about cleaning service |
| **Actors** | Users, Speech to Text Service, Mailing Service, Server |
| **Description** | If a user notices a cleaning issue he/she can notify related workers via Garcon. |
| **Data** | Audial input from user |
| **Preconditions** | User should be authenticated |
| **Stimulus** | User giving audial input about issuing a cleaning request |
| **Basic Flow** | Step 1 – User gives audial input  Step 2 – Audial input gets processed by Speech to Text Service  Step 3 – Issue is created and server side is informed  Step 4 – Processed text of audial input is sent as an email to the related workers  Step 5 – User is informed that issue is registered |
| **Alternative Flow** | Step 3 – System detects same request already issued  Step 4 – Importance level of request is updated  Step 5 – User is informed that issue is already registered |
| **Exception Flow** | - |
| **Post conditions** | An issue instance is created on system and related workers are informed. |

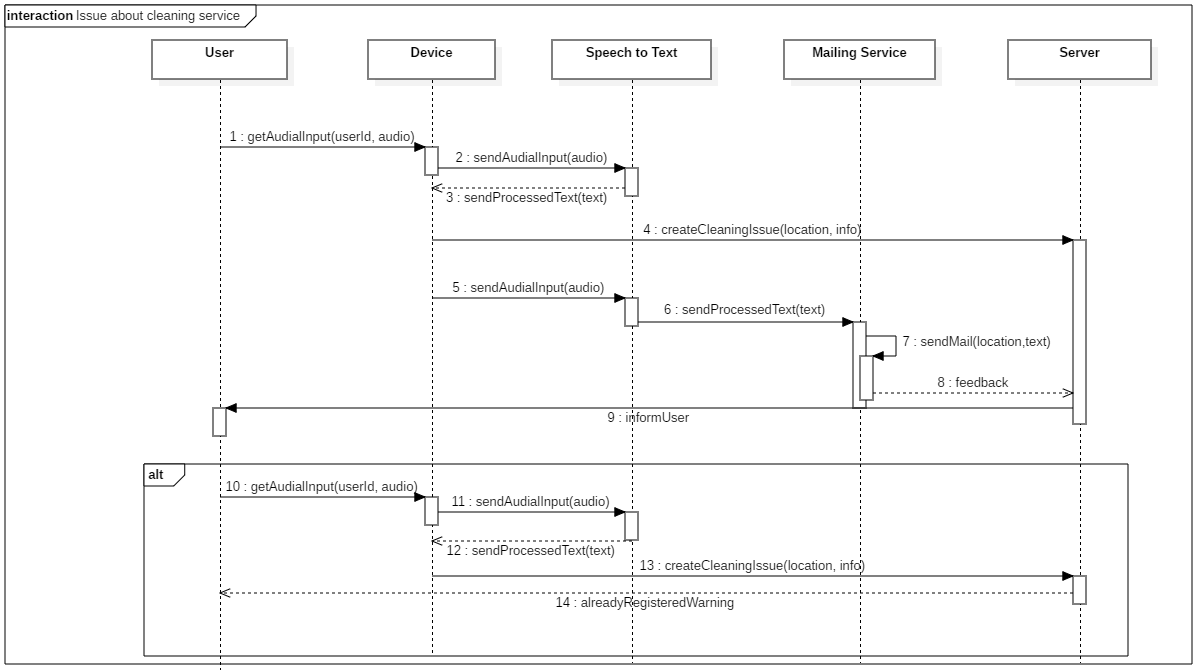


Table 2: Open cleaning issue function

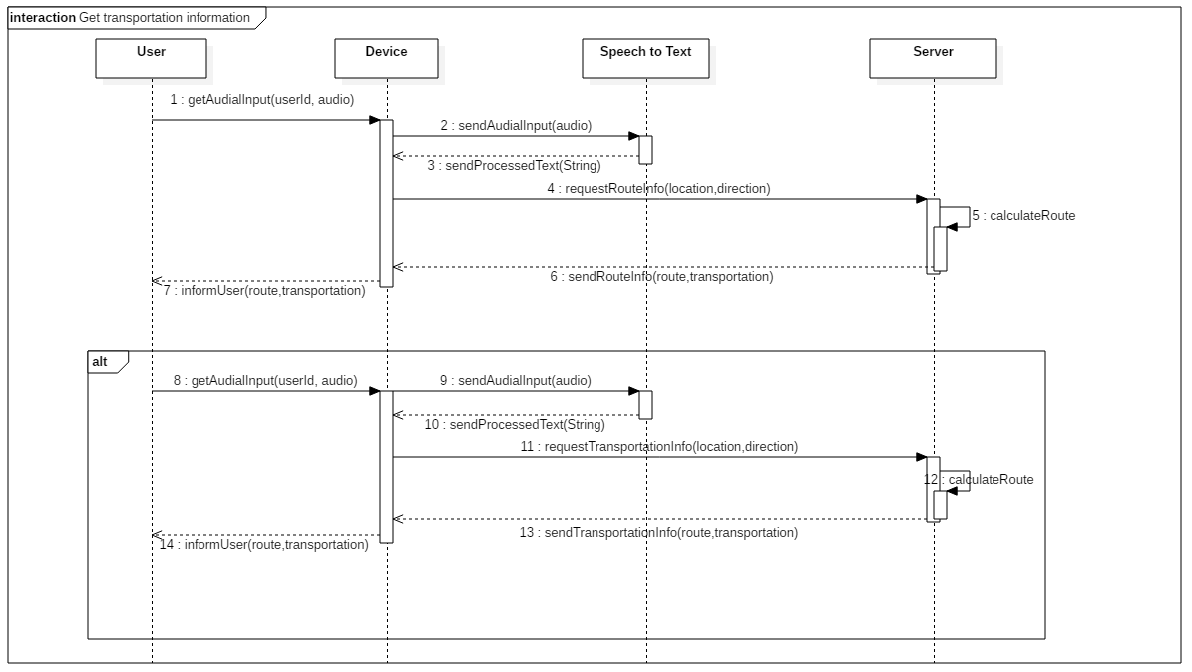
**Figure 4: Issue about cleaning service Sequence Diagram**

|  |  |
| --- | --- |
| **Use case name** | Order food from Yemeksepeti |
| **Actors** | Users, Speech to Text Service, Yemeksepeti API |
| **Description** | When user asks for ordering food, the request translated into text first, gets analyzed and then system automatically give an order from Yemeksepeti |
| **Data** | Audial input from user |
| **Preconditions** | User should be authenticated |
| **Stimulus** | User giving audial input about ordering food |
| **Basic Flow** | Step 1 – User gives audial input  Step 2 – Audial input gets processed by Speech to Text Service  Step 3 – A request is posted to Yemeksepeti Api  Step 4 – Success message shown to the user |
| **Alternative Flow** | - |
| **Exception Flow** | If any error occurs or restaurant is closed , system will show a log message. |
| **Post conditions** | User gives an order from Yemeksepeti. |

Table 3: Order food function

|  |  |
| --- | --- |
| **Use case name** | Get transportation information |
| **Actors** | Users, Speech to Text Service, Server |
| **Description** | Information service on transformation info. User can get a possible route to a direction from the device he/she is now interacting with and user can get information about transportation schedules. |
| **Data** | Audial input from user, current transportation services’ data, map data |
| **Preconditions** | Worker should be authenticated |
| **Stimulus** | User giving audial input about getting informed on transportation |
| **Basic Flow** | Step 1 – User gives audial input asking directions  Step 2 – Audial input gets processed by Speech to Text Service  Step 3 – Best possible route and its transformation information is obtained from the server  Step 4 – Information is converted to audio form by Speech to Text Service  Step 5 – User is informed |
| **Alternative Flow** | Step 1 – User gives audial input asking transportation schedules  Step 2 – Audial input gets processed by Speech to Text Service  Step 3 –Transformation information on ring, bus, subway services is obtained from the server  Step 4 – Information is converted to audio form by Speech to Text Service  Step 5 – User is informed |
| **Exception Flow** | - |
| **Post conditions** | User is informed with best routes and transportation information and information about this query is saved to database to inform further queries faster. |

Table 4: Get transportation info function



**Figure 5: Get transportation information Sequence Diagram**

|  |  |
| --- | --- |
| **Use case name** | Get campus events informations |
| **Actors** | Users, Speech to Text Service, Server |
| **Description** | When user asks for available events in campus, server interacts the database and submits events for user’s information. |
| **Data** | Audial input from user |
| **Preconditions** | User should be authenticated |
| **Stimulus** | User giving audial input about campus event informations |
| **Basic Flow** | Step 1 – User gives audial input  Step 2 – Audial input gets processed by Speech to Text Service  Step 3 – Available events searched in server  Step 4 – Available events are converted to audio format.  Step 5– Events listed to the user. |
| **Alternative Flow** | Step 4 – If no event is available on campus, Garcon will not give any listings. |
| **Exception Flow** | If any error occurs, system will show an error message |
| **Post conditions** | System shows all events |

Table 5: Get event information function

|  |  |
| --- | --- |
| **Use case name** | Close an issue |
| **Actors** | Workers, Speech to Text Service, Mailing Service, Server |
| **Description** | Workers close issues they have handled. |
| **Data** | Audial input from worker |
| **Preconditions** | Worker should be authenticated |
| **Stimulus** | Worker giving audial input about closing an issue |
| **Basic Flow** | Step 1 – Worker gives audial input  Step 2 – Audial input gets processed by Speech to Text Service  Step 3 – Issue is closed and server side is informed  Step 4 – Worker is informed that issue is closed |
| **Alternative Flow** | - |
| **Exception Flow** | - |
| **Post conditions** | The issue is marked as solved on system. |

Table 6: Close issue function

|  |  |
| --- | --- |
| **Use case name** | Register to an issue |
| **Actors** | Workers, Speech to Text Service, Server |
| **Description** | Workers register to an issue to prevent possible conflicts. |
| **Data** | Audial input from worker, current registered workers on the issue |
| **Preconditions** | Worker should be authenticated |
| **Stimulus** | Worker giving audial input about registering to an issue |
| **Basic Flow** | Step 1 – Worker gives audial input  Step 2 – Audial input gets processed by Speech to Text Service  Step 3 – Current workers on issue are displayed to worker  Step 4 – Worker is asked a confirmation after seeing current workers on issue  Step 5 – With workers confirmation he is registered to issue (Database update) |
| **Alternative Flow** | - |
| **Exception Flow** | If worker does not confirm after seeing current workers on the issue the process is aborted. |
| **Post conditions** | The worker is registered to issue. |

Table 7: Register issue function

|  |  |
| --- | --- |
| **Use case name** | Getting informations about issues |
| **Actors** | Workers, Speech to Text Service |
| **Description** | Worker scans his/her id card and Garcon gets activated. Then waits for worker to talk to decide what to do. |
| **Data** | Audial input from worker |
| **Preconditions** | Worker should be authenticated |
| **Stimulus** | Worker giving audial input about open issues |
| **Basic Flow** | Step 1 – Worker gives audial input  Step 2 – Audial input gets processed by Speech to Text Service  Step 3 – Server returns open issues from Database |
| **Alternative Flow** | - |
| **Exception Flow** | - |
| **Post conditions** | Worker can see the whole available/open issues from database. |

Table 8: Get issue information function

|  |  |
| --- | --- |
| **Use case name** | Authentication |
| **Actors** | Users, Workers, Admins, Card Reader |
| **Description** |  |
| **Data** | Chip from Id Cards |
| **Preconditions** | - |
| **Stimulus** | Id Card must be scanned |
| **Basic Flow** | Step 1 – User/Worker/Admin holds Id Card to the device  Step 2 – Device reads the Card and authenticate |
| **Alternative Flow** | - |
| **Exception Flow** | Step 1 – Card doesn’t recognized or cannot be scanned  Step 2 – System displays a visual error output |
| **Post conditions** | Authenticate successful and device gets waiting for audial input to work. |

Table 9: Authentication

|  |  |
| --- | --- |
| **Use case name** | Adding users |
| **Actors** | Admin |
| **Description** | Admin adds new user to database |
| **Data** | - |
| **Preconditions** | Admin should be authenticated |
| **Stimulus** | Admin giving information about new user |
| **Basic Flow** | Step 1 – Admin gives information about user  Step 2 – Input gets processed by Speech to Text Service  Step 3 – User Id Card Scanned  Step 4 – New user added |
| **Alternative Flow** | - |
| **Exception Flow** | Step 1 – Admin gives information about user  Step 2 – Input cant be recognized  Step 3 – System gives a warning |
| **Post conditions** | New user added to the database. |

Table 10: Add user function

|  |  |
| --- | --- |
| **Use case name** | Blocking users |
| **Actors** | Admin |
| **Description** | Admin blocks user from database |
| **Data** | - |
| **Preconditions** | Admin should be authenticated |
| **Stimulus** | Admin gives delete command and gives information |
| **Basic Flow** | Step 1 – Admin gives information about user to be deleted  Step 2 – Input gets processed by Speech to Text Service  Step 3 – User deleted |
| **Alternative Flow** | - |
| **Exception Flow** | Step 1 – Admin gives information about user  Step 2 – Input cant be recognized  Step 3 – System gives a warning |
| **Post conditions** | User deleted from the database. |

Table 11: Block user function

## 3.3 Usability Requirements

● Any request for information or issues are creating with audial input of the user.

● Every conversation with Garcon should start with scanning ID Card.

● Every conversation ends with an auto-message generated by Garcon.

● For every unrecognized audio input , Garcon will generate an audio error message for user to ask another request.

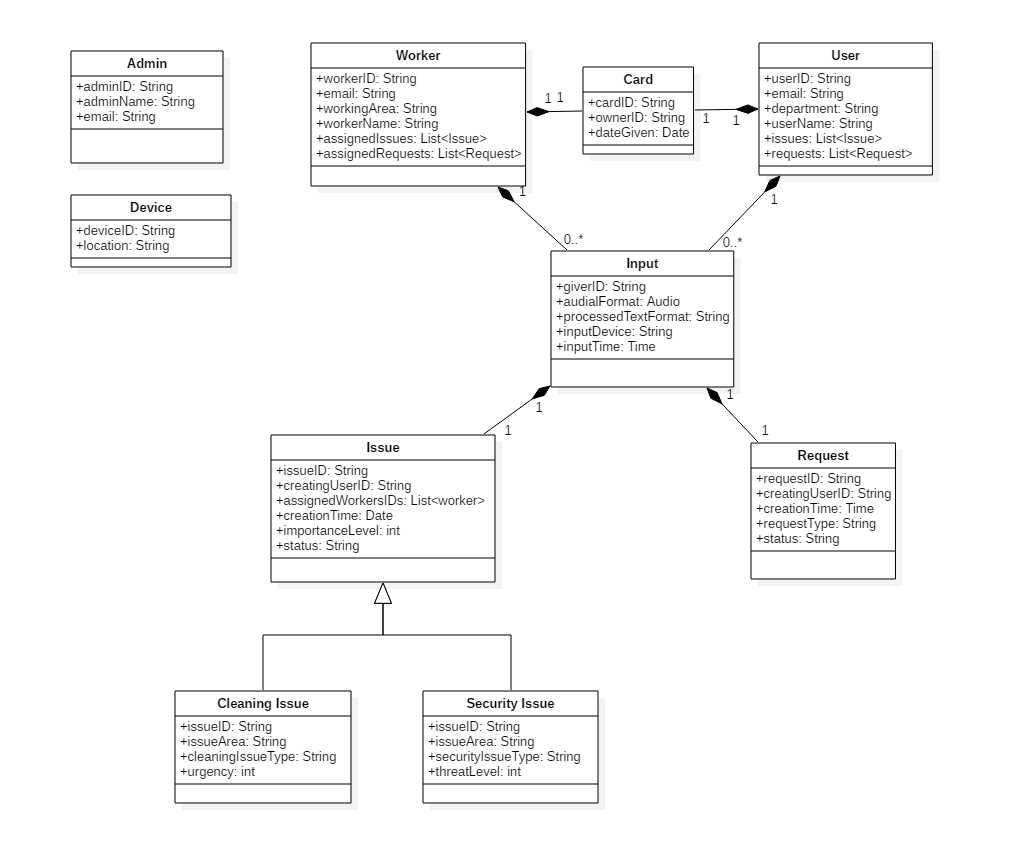
## 3.4 Performance Requirements

● Garcon should return respond to user in at most 4 seconds.

● Internet speed should be enough at least 2Mbps to maintain the system.

● Number of simultaneous users to be supported should be 1000 users.

## 3.5 Logical Database Requirements



**Figure 6: Logical Database Requirements Class Diagram**

* Only the admin is able to register/block users/workers.
* Card entities are added by admins and are given only one per user/worker.
* Card table is only accessible by admins.
* User and worker entries added/deleted by operations of admins.
* User/worker tables are only accessible by admins.
* User/worker-card tables has one to one relationship since one user/worker can own only one card in our system.
* Input entries are created when user/worker scan his/her card then talks.
* After processing inputs system may create issue entries or request entries.
* User/worker-input tables has one to many relationship since one user/worker can give multiple inputs.
* An issue is either a cleaning issue or a security issue.
* Issue/Request entries cannot be added without a corresponding input entry.
* When an issue is closed its entry is not deleted from database, just its status is updated to “closed”.
* When a request is fulfilled its entry is not deleted from database, just its status is updated to “fulfilled”.
* Device table can be accessed only by admins.
* Admins can only be registered to system by other admins.

## 3.6 Design Constraints

All data must be kept private and stored for legal purposes.

## 3.7 Software System Attributes

**a) Reliability:** Possible errors or system failures should be logged and admins should maintain the system out of working hours when least amount of users are using system.

**b) Availability:** System should be available all the time.

**c) Privacy & Security:** All requests done by users and all informations about users should be kept private.

**d) Maintainability:** System should be designed in a way that eases maintenanceand documentation should be well written.

**e) Portability:** System should be available in mostly used places on campus.There is no need for mobile/web app since users have a nearby iot device all the time.

## 3.8 Supporting Information

# Verification

# Appendices

## Assumptions and Dependencies

## Acronyms and Abbreviations