<inteliQ>

Software Requirements Specification

Version <1.1>

Revision History

Date	Version	Description	Author
<MM/DD/YYYY>	<version>	<description>	<author>

Table of Contents

1. Introduction

Our software aims at creating an intelligent forms' answering application where each selected option of a question leads to a specific next question. Our software consists of a frontend application for the users of our application and a CLI that can be used by an administrator to manage the system.

Each user has the potential to either answer a specific questionnaire (role : *participant*) or create a new questionnaire for other users to answer (role : *admin*). The administrator of a questionnaire can retrieve statistical data concerning the answers given.

1.1 Overall Description

1.1.1 Problem Statement

Creation of a questionnaire that can be dynamically configured as each user is answering it.

1.1.2 Objectives

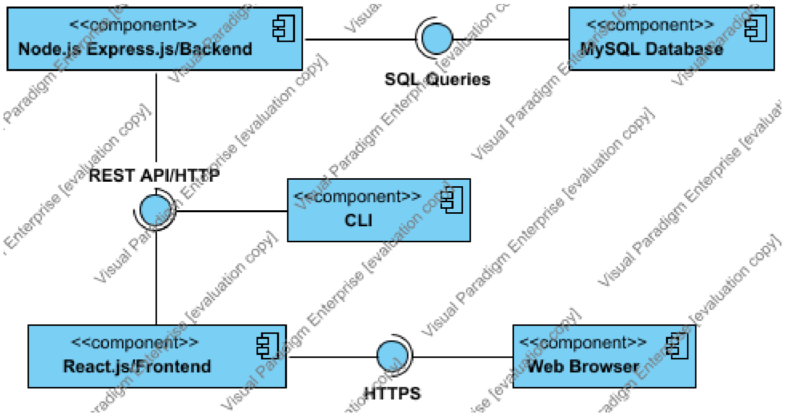
A convenient and easy to use User Interface in order to create and answer questionnaires and a CLI so that the System Administrator can easily manage the application.

1.2 Interfaces

1.2.1 Interfaces with internal systems

- A React Web Server used for the frontend
- A Rest API used for the communication between the backend and the frontend or the CLI
- MySQL Database used for storing data.

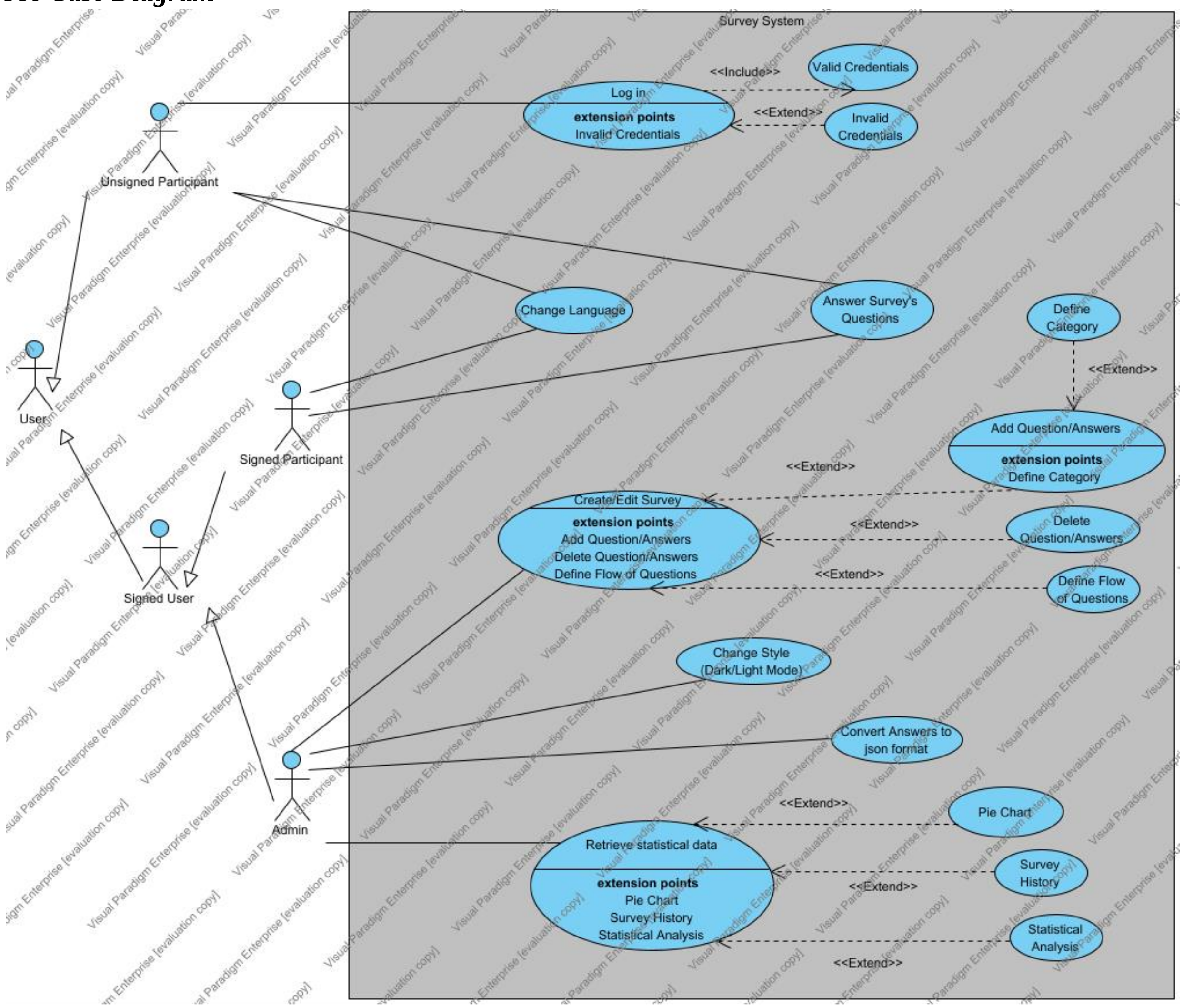
Component Diagram



1.2.2 Interfaces with Users

Below we can see the Use Case Diagram which describes all possible use cases with the respective roles/actors.

Use Case Diagram



1.3 References

N/A

2. Functional Requirements – Use Cases

Below, we will describe two possible use cases.

2.1 Use Case 1 : Answer Survey’s Questions

2.1.1 Roles

Participant in a specific questionnaire.

2.1.2 Prerequisites

Logged in as a user

2.1.3 Execution Environment

The participant can answer the questionnaire using the frontend Web Server which in its turn communicates with the backend server and the Database to render the Website and provide the user with the necessary data. In this case, no extra communication with an External Server is needed.

2.1.4 Input Data

Firstly, each user has to fill the required personal data information in order to proceed with the questionnaire.

Each User chooses the desired answer in each question and then presses the <Next> Button in order to answer the next question his answer will lead to until the questionnaire is fully filled.

2.1.5 Parameters

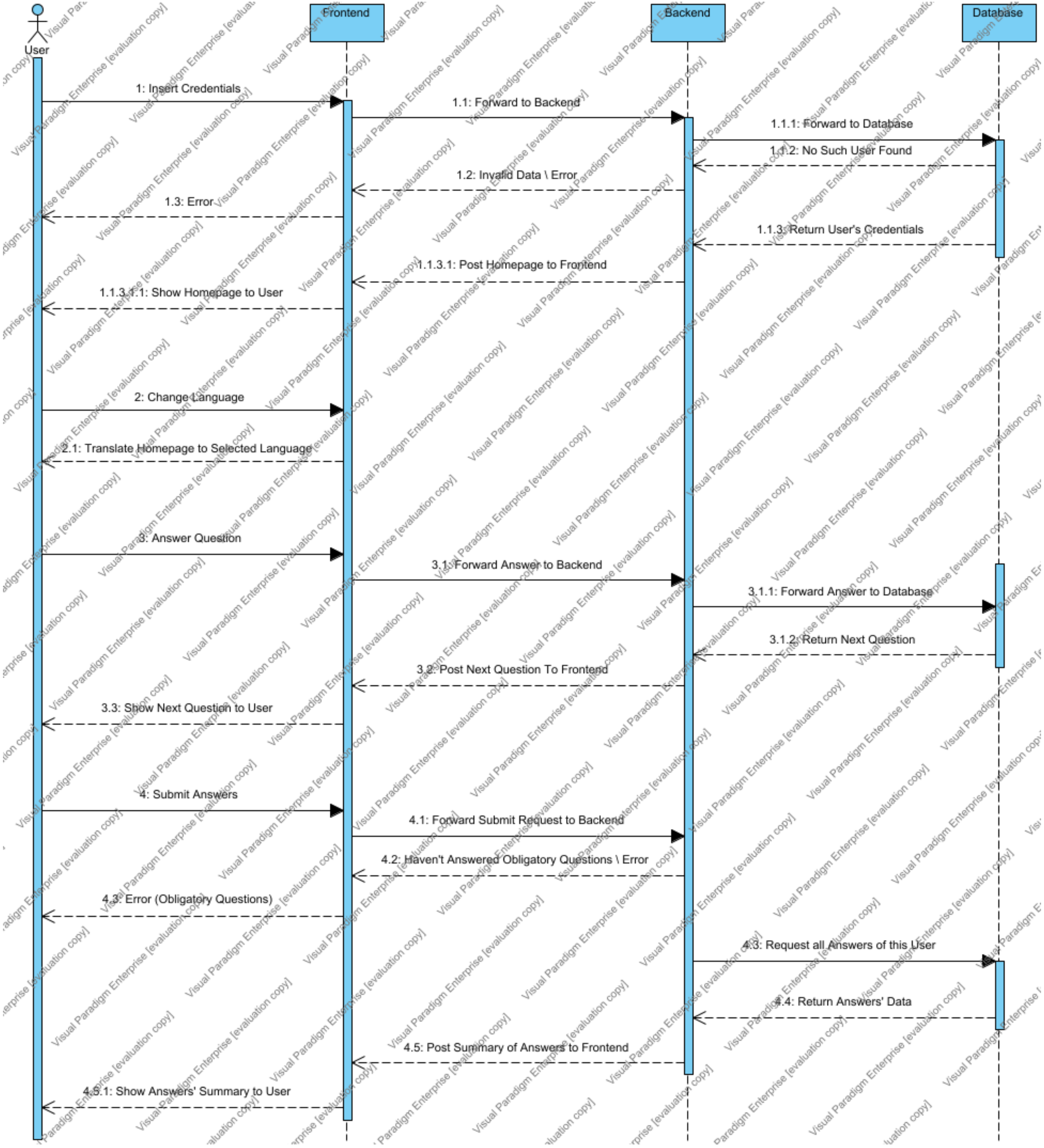
After the use begins answering the questionnaire, the following parameters must be transported from the frontend to the backend for further processing :

- Questionnaire Information : Transfer the ID of the questionnaire the user is currently answering
    - ✓ questionnaireID : String
- Question Information : Make Backend aware of the question that has been answered and the answer that has been given in the current session.
    - ✓ qID : String
    - ✓ ans : String
    - ✓ session : String

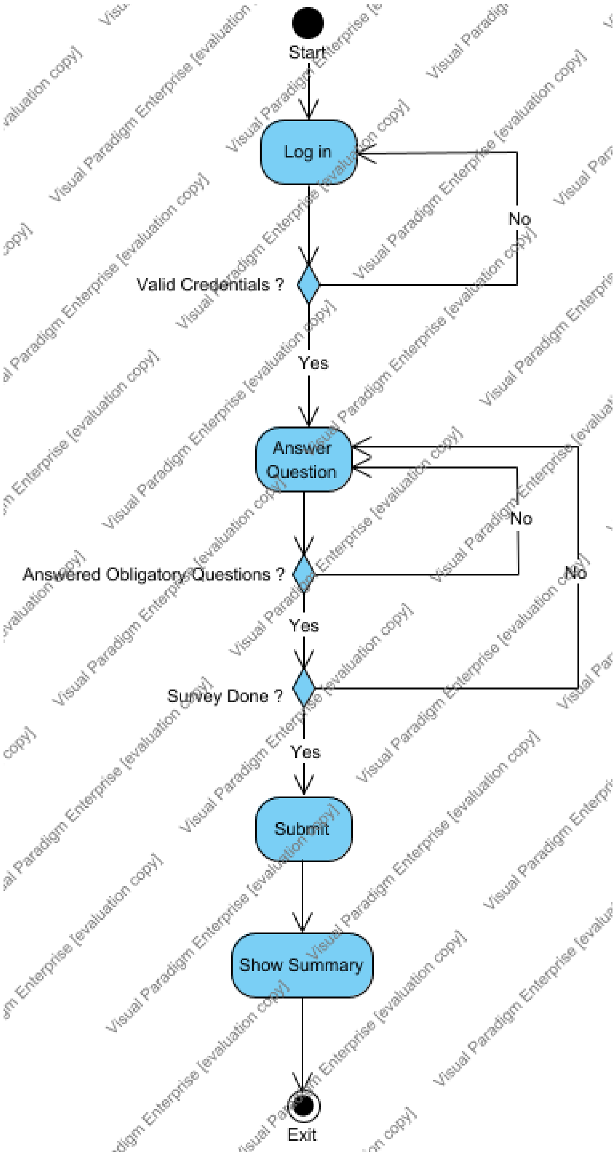
2.1.6 Sequence of Actions – Sequence/Activity Diagram

Below we can see the Sequence and Activity Diagrams for this Use Case.

User - Sequence Diagram


User
Frontend
Backend
Database
1: Insert Credentials
1.1: Forward to Backend
1.1.1: Forward to Database
1.1.2: No Such User Found
1.2: Invalid Data \ Error
1.3: Error
1.1.3: Return User's Credentials
1.1.3.1: Post Homepage to Frontend
1.1.3.1.1: Show Homepage to User
2: Change Language
2.1: Translate Homepage to Selected Language
3: Answer Question
3.1: Forward Answer to Backend
3.1.1: Forward Answer to Database
3.1.2: Return Next Question
3.2: Post Next Question To Frontend
3.3: Show Next Question to User
4: Submit Answers
4.1: Forward Submit Request to Backend
4.2: Haven't Answered Obligatory Questions \ Error
4.3: Error (Obligatory Questions)
4.3: Request all Answers of this User
4.4: Return Answers' Data
4.5: Post Summary of Answers to Frontend
4.5.1: Show Answers' Summary to User

Answer Survey's Questions



2.1.7 Output Data

After each question has been answered, our application must render the next question in the Web Page (i.e. nextqID, qtext, options…) and finally when the whole questionnaire is answered a summary of all selected answers must be shown to the user (i.e. answers …).

2.2 Use Case 2 : Create and Edit Survey

2.2.1 Roles

Administrator of a specific questionnaire.

2.2.2 Prerequisites

Logged in as an admin

2.2.3 Execution Environment

The administrator can create or edit a questionnaire using the frontend Web Server which in its turn communicates with the backend server and the Database to post the newly created questionnaire's data to the Database.

2.2.4 Input Data

Firstly, the admin has to add questions and their respective answers and then determine the flow of the questions according to the selection of answers by the participant. The Admin can also edit a questionnaire of his own by changing a specific question, its answers or the flow.

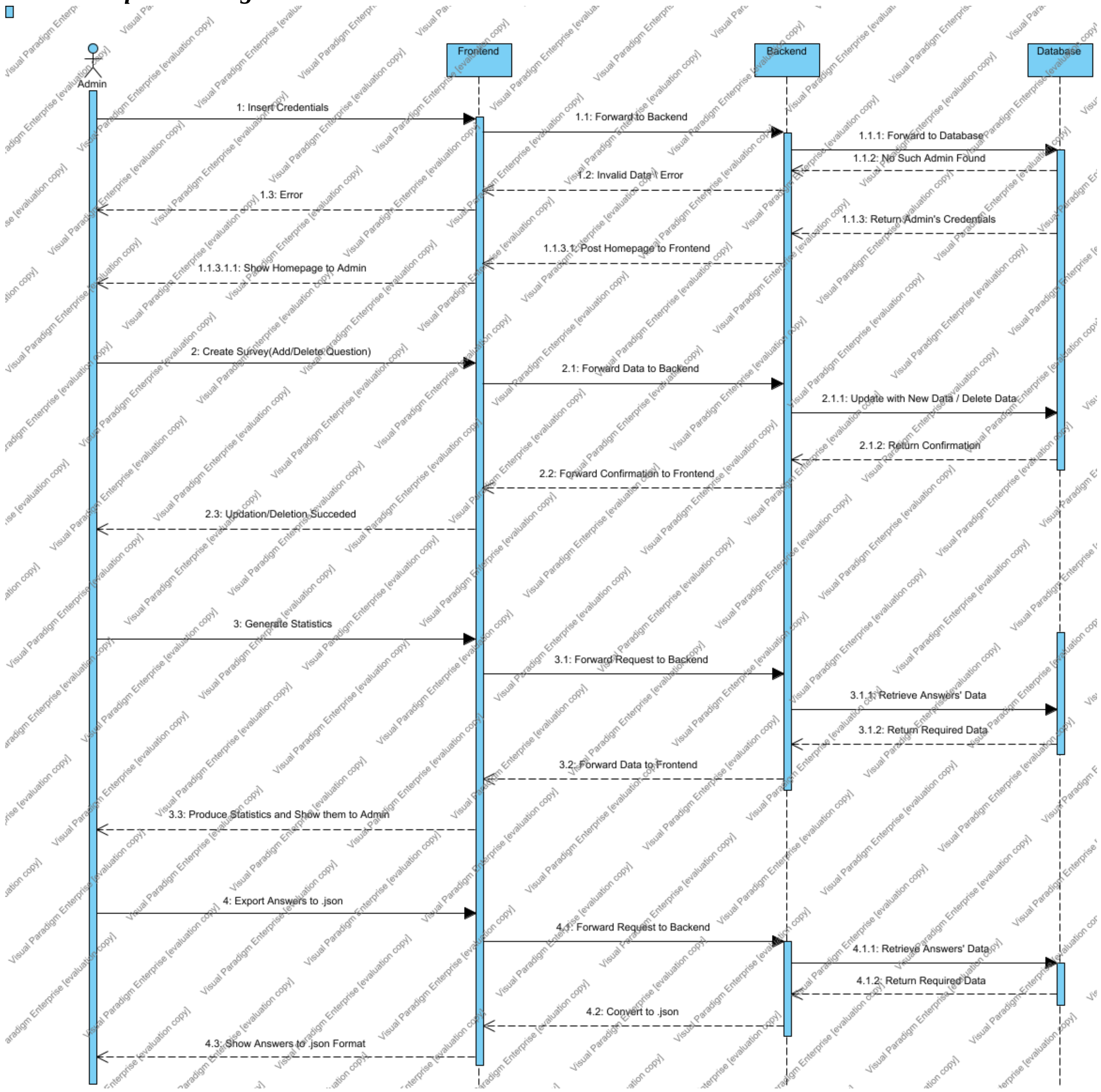
2.2.5 Parameters

- Questionnaire Information : Transfer the ID of the questionnaire the admin created/edited
    - ✓ questionnaireID : String
- Question Information : Make Backend aware of the question and answers that have been added and the determined flow.
    - ✓ qID : String
    - ✓ options : List
    - ✓ nextqID : String

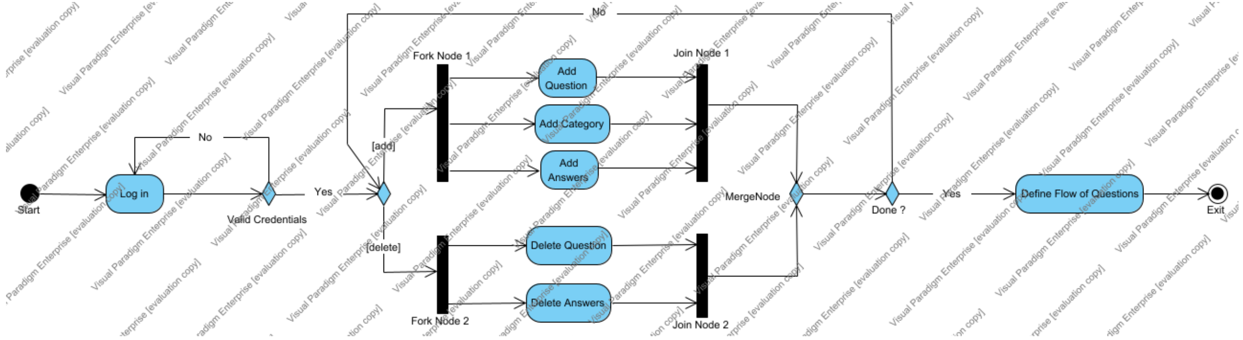
2.2.6 Sequence of Actions – Sequence/Activity Diagram

Below we can see the Sequence and Activity Diagrams for this Use Case.

Admin - Sequence Diagram


Admin
Frontend
Backend
Database
1: Insert Credentials
1.1: Forward to Backend
1.1.1: Forward to Database
1.1.2: No Such Admin Found
1.2: Invalid Data \ Error
1.3: Error
1.1.3: Return Admin's Credentials
1.1.3.1: Post Homepage to Frontend
1.1.3.1.1: Show Homepage to Admin
2: Create Survey(Add/Delete Question)
2.1: Forward Data to Backend
2.1.1: Update with New Data / Delete Data
2.1.2: Return Confirmation
2.2: Forward Confirmation to Frontend
2.3: Updation/Deletion Succeded
3: Generate Statistics
3.1: Forward Request to Backend
3.1.1: Retrieve Answers' Data
3.1.2: Return Required Data
3.2: Forward Data to Frontend
3.3: Produce Statistics and Show them to Admin
4: Export Answers to .json
4.1: Forward Request to Backend
4.1.1: Retrieve Answers' Data
4.1.2: Return Required Data
4.2: Convert to .json
4.3: Show Answers to .json Format

Create/Edit Survey



2.2.7 Output Data

Produce Success or Failure Message

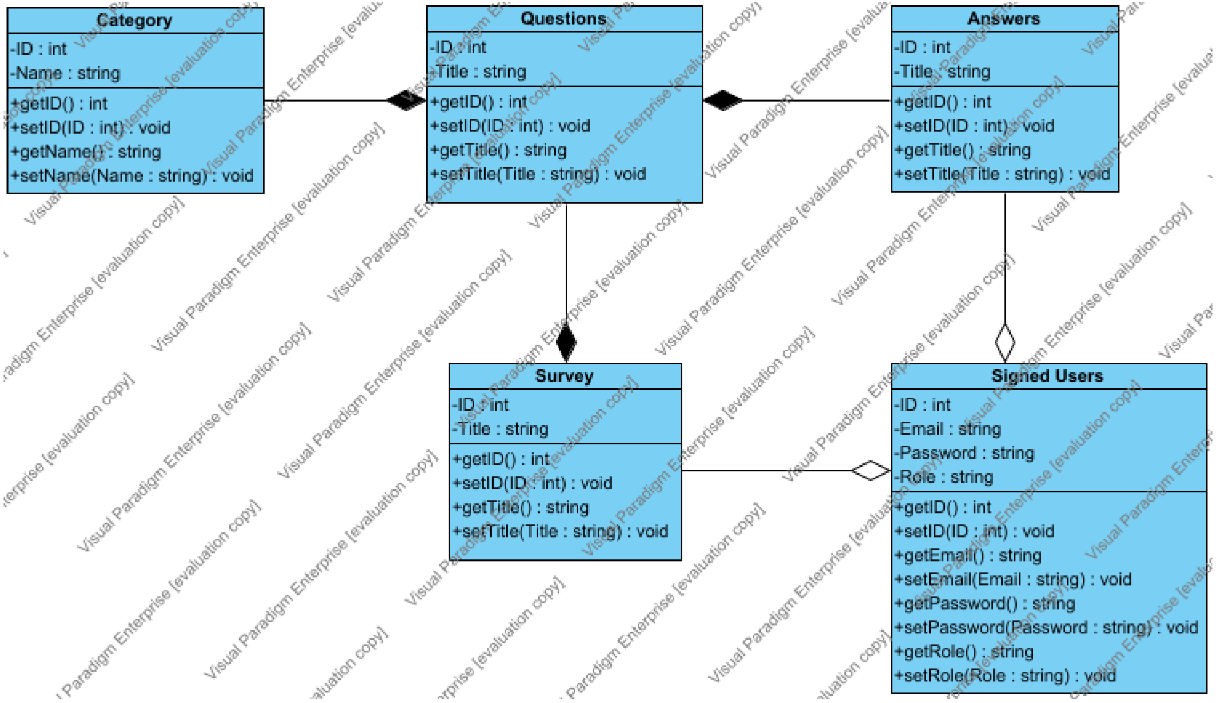
2.3 Performance Requirements

Our software will be available for use throughout each day and we expect for our software to be able to deal with users’ requests (log in or data get/post) without significant delay.

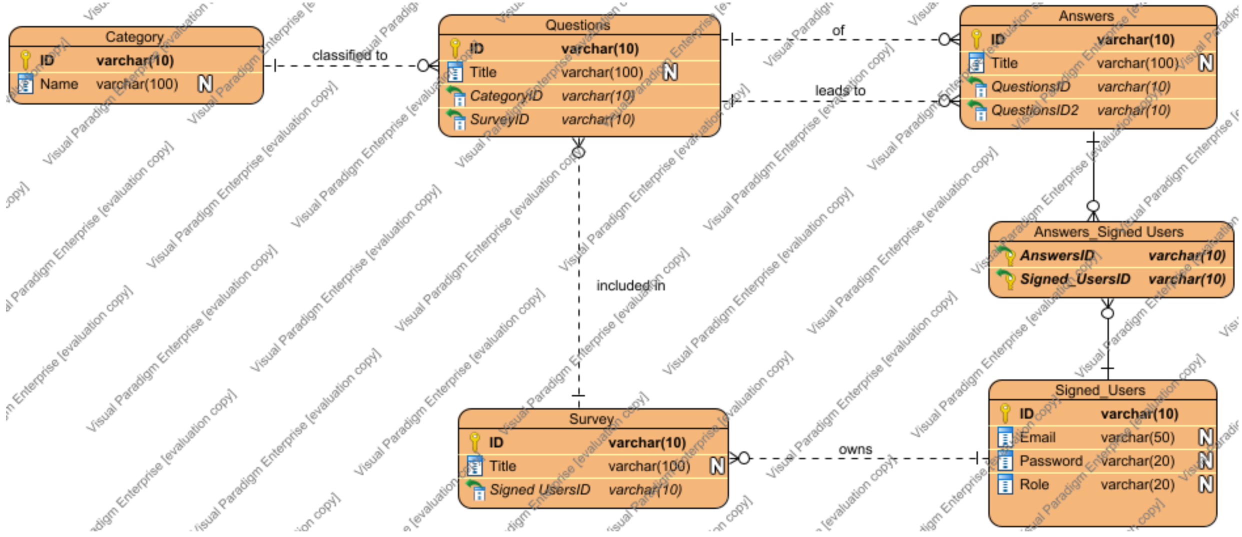
2.4 Data Organization Requirements

Below, we can see the Class Diagram of our System and the Entity Relationship Diagram of the Database.

Class Diagram



Entity Relationship Diagram



2.4.1 Data Access Limitations

Users have access to only specific parts of the Database, such as questionnaires, questions and answers and not personal information of other users. Only the System administrator is able to handle data concerning usernames and passwords.

2.5 Design Constraints

<To be Filled>

2.6 Other Requirements

2.6.1 Software Availability

The software has to be available for use throughout the day.

2.6.2 Security

Use of HTTPS encryption in order to avoid data interceptions by third parties so that out users’ data can be kept safe while interacting with our Website. Users’ passwords are stored encrypted in out Database and Users are required to log in, in order to use the Website.

2.6.3 Maintenance

Maintenance of our system needs to be performed regularly and is subject to the number of users our system hosts.