Design document

**Name: Tony Jiang**

**Date: 19 Sep 23**

**Project: Video call system**

|  |  |  |
| --- | --- | --- |
| **Versions** | **Date** | **Description** |
| **0.1** | **19 Sep 23** | **Initial document** |
|  |  |  |

Contents

[Introduction 3](#_Toc146208681)

[Context 3](#_Toc146208682)

[Goals 3](#_Toc146208683)

[Scope 3](#_Toc146208684)

[Functionality 4](#_Toc146208685)

[User Interface Design 4](#_Toc146208686)

[System Architecture 4](#_Toc146208687)

[Level 1: System Context (C1) 4](#_Toc146208688)

[Level 2: Containers (C2) 5](#_Toc146208689)

[Level 3: Component (C3) 5](#_Toc146208690)

[Level 4: Code (C4) 5](#_Toc146208691)

# Introduction

This document covers all the technical aspects of this project, including its structure and the design decisions made throughout the development. It helps to further reflect on the design choices within this project. By doing so, it helps with planning how everything should be configured and address any potential development challenges. Additionally, it helps in conveying the intended design to other developers, ensuring a shared understanding and agreement on the design approach. Some information may refer to other documents where the information is fully detailed. New content or components will be added to this document throughout the project, and it is intended for technical developers.

The project's objective is to create two or three solutions for a video call system within the PRAS application.

# Context

This is to give general information about the project. What’s the project about and from whom. For context, please refer to the **Project Plan** document, on ‘*1.1 Context*’.

# Goals

This is to give information about what the project is trying to achieve and what the purpose of the project is. For the goals of this project, please refer to the **Project Plan** document, on ‘*1.2 Goal of the project*’.

# Scope

This is to define what needs to be delivered for the entire project, to ensure clarity for all stakeholders. The scope of the project can be found in the **Project Plan** document, on ‘*1.3 Scope and preconditions’*.

# Functionality

This is to define the core functionality of the system, so that it’s clear to everyone. To achieve a clear understanding of the functionality of the system, we use something called ‘user story’. User story describes the functionality of the product, the expected behavior, and the required components. You can find the user stories in the **User Story** document.

# User Interface Design

The purpose of this is to present the user interface to all the stakeholders and reach an agreement on the product’s visual design. To illustrate this, we made use of wireframes, which are mockup designs of an application or website. Think of wireframes as the initial sketch when you're creating a painting; they lay out what will appear on the canvas before you start painting. There are websites or applications that allow you to easily create wireframes for free. You can create, export, demonstrate, and get feedback on it instantly. With the wireframe you can also provide detailed descriptions for each component on the wireframe. The wireframes can be found in the **Wireframe document**. Tools for creating the wireframe: [Figma](https://www.figma.com/files/recents-and-sharing/recently-viewed?fuid=1217416791179059026).

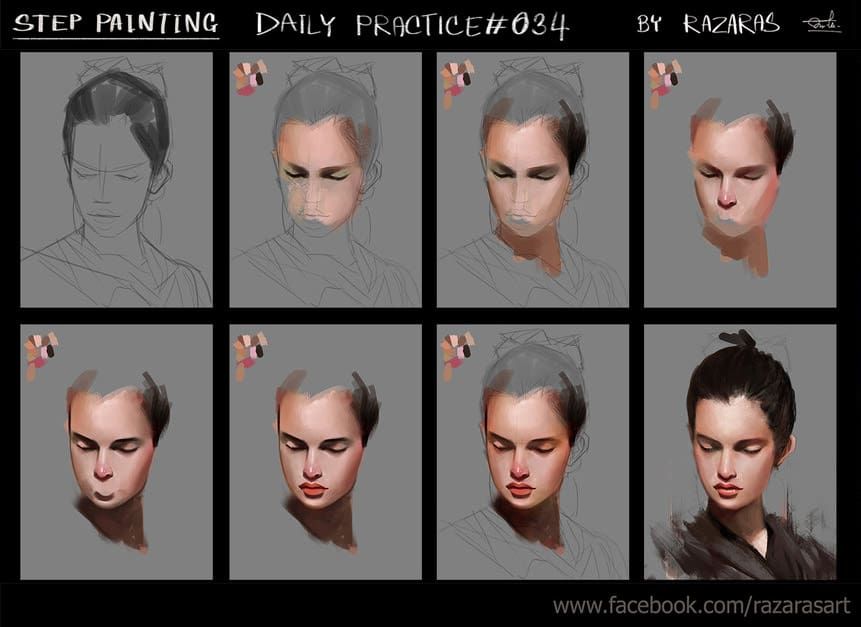


Figure : Sketch painting example

# System Architecture

The purpose of System Architecture is to describe the internal system’s overall structure and establish an agreement on the desired design of the system. To make it easy to describe and communicate the system’s architecture, we’ll use a C4 architecture diagram. It’s an architecture design that is easy to understand. The design approach is straightforward and helps us communicate how each part of the system should be set up, even to a non-technical person. It’s like using Google Maps and you’re trying to zoom in on a place in Aruba. The closer you zoom the more details you see. It’s like that. The C4 architecture diagram helps us do the same for our software’s structure.

The C4 architecture diagram has 4 levels:

* Level 1: System Context (C1)
* Level 2: Containers (C2)
* Level 3: Component (C3)
* Level 4: Code (C4)

We will go to each level and describe what they do when we reach there. The tool that was used to create the C4 model is [Visual Paradigm online](https://online.visual-paradigm.com/diagrams/features/c4-model-tool/). The free version.

## Level 1: System Context (C1)

At this level you’ll discover which intended user is going to use the system and what the system does. This provides an overview of the entire system. Additionally, it helps to describe what the system does to non-technical people.

A screenshot of a computer

Description automatically generated

Figure : C1 model

In the C1 model, you can see that employees can make video calls to retirees using the video call system. This provides an understanding of which users can use the system, what kind of system it is and the action they perform on the system.

## Level 2: Containers (C2)

At this level, we delve deeper into the system’s architecture, moving beyond the high-level system context of level 1. Here, we identify the high-level technical building blocks or ‘containers’ that make up the system and understand the relationship between them. These containers represent the applications or databases used to build the entire system. It’s important to note that the information presented from this level onwards is intended for the technical audience.

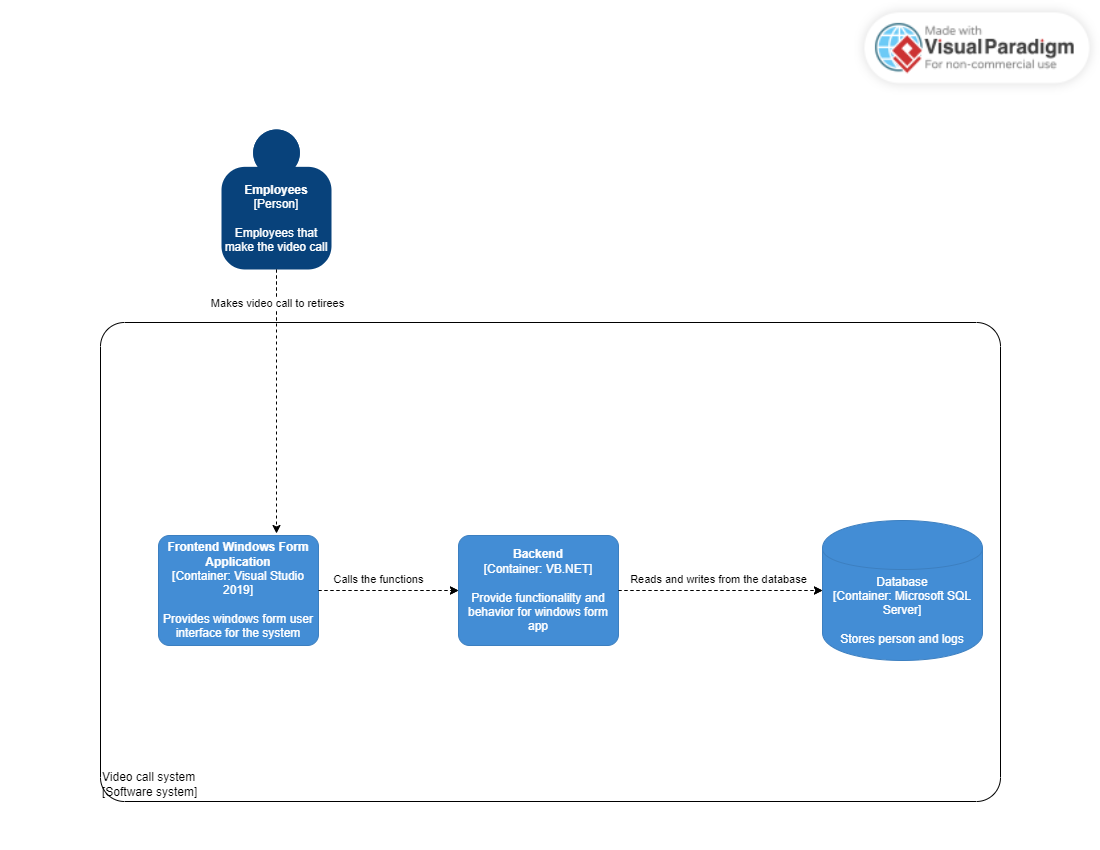


Figure : C2 model

It’s pretty understandable how the system is put together from the C2 model. There are 3 containers that make up the system.

* **Frontend Windows Form Application**
* This component serves as the user interface, where the users interact with the system.
* It communicates with the backend to process user’s actions.

**Technology: Visual Studio 2019**

* The PRAS application is developed in Visual Studio 2019
* The PRAS application uses Windows Form App
* It ensures seamless integration within the same IDE for the video call system.
* **Backend**
* "All the functional code responsible for the system’s operations, resides in here.
* It manages data flow between the front end and the database.

**Technology: Visual Basic .NET (VB.NET)**

* The backend is written in VB.NET language for the PRAS application.
* Utilizing the same language for simpler integration into the PRAS application.
* **Database**
* This container stores all the retirees’ information and video call logs.
* It provides data to the backend as needed.

**Technology: Microsoft SQL Server**

* The PRAS application relies on Microsoft SQL Server as its database system.
* Utilizing the PRAS application’s existing database simplifies the process of fetching and adding data and avoid the necessity of making a new database and populate it.

## Level 3: Component (C3)

Level 3 represents the component (C3) of the container from level 2 of the system’s architecture. These components are the building blocks that make up the containers of level 2 and they interact with each other. These components are categorized by the function they are assigned to do. For example, policemen are tasked to arrest criminals, you can’t task a cook to do that. It isn’t their job to do it.

A diagram of a company

Description automatically generated

Figure : C3 model from Frontend Windows Form Application

This is the C3 model of the ‘Frontend Windows Form Application’ container. It contains 2 components. The design idea here is to sperate the user interfaces with destinct functions.

* Video Call Presentation Layer: Here lies all the video call user interfaces
* Log Presentation Layer: This contains all the logs related to the user interfaces.

A diagram of a company

Description automatically generated

Figure : C3 model from backend container

This diagram shows structure of the Backend container, which consist of 4 components that do their own task. This separation of task simplifies the understanding of the design and for future additional component.

* Video Call Business Layer
* Contains all the video call functions.
* It uses the ‘Person Data Access Layer’ to retrieve retiree data.
* It uses the ‘Logs Data Access Layer’ to add logs to the database.
* Logs Business Layer
* Contains all the logs functions.
* It uses the ‘Log Data Access Layer’ to retrieve and add log data.
* Person Data Access Layer
* Manages data flows from the database to specific video call functions.
* Logs Data Access Layer
* Manages data flows from the database to specific log functions.

Level 4: Code (C4)

This level contains UML diagrams for each component, providing detailed information on the relationships between the classes. This level is typically optional, it used mainly to illustrate complex class structures and explain how it works. This is going to be used for conveying how the video call system is structure in the backend.

Testing strategies

This is to plan out how I’m going to test the video call system, what type of strategy I’m going to use for testing, and the reasons for the tests. For these tests, you would need to follow a step-by-step instruction on how to test the system. The summarize version of testing strategies is in the project plan document in section 4.1, ‘Testing strategies’.

Test approach

|  |  |  |  |
| --- | --- | --- | --- |
| Test type | Explanation | When should you start | By whom? |
| Unit test | To test the functionality of the system and ensure that everything is working as expected, even after new line of code are added. In order to prevent any code error. | This should start at the initial documentation phase. To get a clear idea of the possible problem that may occur to the application. | By Tony |
| User acceptance test | To test the expected user outcomes and prevent any unexpected results to show on the system. | At the final phase of testing everything | By Tony |

Unit test

In here we will be defining all the unit test that needs to be written and tested on the system.

**Method for getting phone number**

ShouldBeAbleToGetPersoonPhoneNumber()

ShouldGiveAnExceptionWhenThereAreNotAnyPhoneNumber()

**Method for calling the video call**

ShouldInitializeVideoCall()

**Method for ending video call**

ShouldEndVideoCall()

**Method for getting email**

ShouldBeAbleToGetPersoonEmail()

ShouldGiveAnExceptionWhenThereIsNotAnyEmail()

**Method for adding date and time of the log**

ShouldBeAbleToAddDateAndTimeOfLog()

**Method for getting the date and time from the video call**

ShouldBeAbleToGetTheDateAndTimeOfVideoCall()

**Method for microphone**

ShouldBeAbleToMuteMic()

ShouldGiveExceptionWhenMicNotFound()

**Method for camera**

ShouldBeAbleToTurnCameraOff()

ShouldGiveExceptionWhenCameraNotFound()

User acceptance test

Here will reside all the user acceptance test and will be tested after all the function are done.