2/14/2025

(Student C00270632) Qadeer Hussain Supervisor: Paul Barry

Software DEVELOPMENT PROJECT YEAR 4

Design Document

Elderly Care Management System(Care Net)

A logo with black text

Description automatically generated

**Table of Contents**

[**Introduction** 3](#_Toc190566126)

[**System Architecture** 4](#_Toc190566127)

[**Technologies & Tools** 5](#_Toc190566128)

[**System Sequence Diagrams** 7](#_Toc190566129)

[**Core Features** 7](#_Toc190566130)

[**Key Server** 7](#_Toc190566131)

[**Patient Profile** 8](#_Toc190566132)

[**Medical Dashboard** 12](#_Toc190566133)

[**Roster** 17](#_Toc190566134)

[**Care Planner** 18](#_Toc190566135)

[**Non Core Features** 20](#_Toc190566136)

[**Class Diagram** 21](#_Toc190566137)

[**Prototype GUI Screens** 22](#_Toc190566138)

[**Main Menu** 22](#_Toc190566139)

[**Patient Profile Prototype** 22](#_Toc190566140)

[**Medical Dashboard** 23](#_Toc190566141)

[**Medical Dashboard Log** 23](#_Toc190566142)

[**Roster** 24](#_Toc190566143)

[**Care Planner Admin** 24](#_Toc190566144)

[**Care Planner Carer** 25](#_Toc190566145)

[**Database Design** 26](#_Toc190566146)

[**Key Server Database** 26](#_Toc190566147)

[**Key Management Table** 26](#_Toc190566148)

[**Key Server Api Token Table** 26](#_Toc190566149)

[**Care Net Database** 26](#_Toc190566150)

[**Patient Profile Table** 26](#_Toc190566151)

[**Users Table** 26](#_Toc190566152)

[**Care Planner Table** 27](#_Toc190566153)

[**Medical Dashboard Table** 27](#_Toc190566154)

[**Roster Table** 27](#_Toc190566155)

[**Scheduling System Table** 27](#_Toc190566156)

[**ER Diagram** 28](#_Toc190566157)

[**Conclusion** 29](#_Toc190566158)

[**References** 30](#_Toc190566159)

**Figure Table**

[Figure 1: System Architecture 4](#_Toc190566160)

[Figure 2: Key Server Token and Key Generation System Sequence Diagram 7](#_Toc190566161)

[Figure 3: Create Patient Profile System Sequence Diagram 8](#_Toc190566162)

[Figure 4: Search Patient Profile System Sequence Diagram 9](#_Toc190566163)

[Figure 5: Edit Patient Profile Sequence Diagram 10](#_Toc190566164)

[Figure 6: Delete Patient Profile System Sequence Diagram 11](#_Toc190566165)

[Figure 7: Search Patient Medical info on Dashboard System Sequence Diagram 12](#_Toc190566166)

[Figure 8: Add Patient Medical info on Dashboard System Sequence Diagram 13](#_Toc190566167)

[Figure 9: Edit Patient Medical info on Dashboard System Sequence Diagram 14](#_Toc190566168)

[Figure 10:Delete Patient Medical info on Dashboard System Sequence Diagram 15](#_Toc190566169)

[Figure 11: Patient Medical Log Dashboard 16](#_Toc190566170)

[Figure 12: Roster System Sequence Diagram 17](#_Toc190566171)

[Figure 13: Add New Care Plan 18](#_Toc190566172)

[Figure 14: Delete Care Plan 19](#_Toc190566173)

[Figure 15: Class Diagram 21](#_Toc190566174)

[Figure 16: Main Menu 22](#_Toc190566175)

[Figure 17: Patient Profile 22](#_Toc190566176)

[Figure 18: Medical Dashboard 23](#_Toc190566177)

[Figure 19: Medical Dashboard Log 23](#_Toc190566178)

[Figure 20: Roster 24](#_Toc190566179)

[Figure 21: Care Planner Admin 24](#_Toc190566180)

[Figure 22: Care Planner Carer 25](#_Toc190566181)

[Figure 23: Key Management Table 26](#_Toc190566182)

[Figure 24: API Token Table 26](#_Toc190566183)

[Figure 25: Patient Profile Table 26](#_Toc190566184)

[Figure 26: Users Table 26](#_Toc190566185)

[Figure 27: Care Planner Table 27](#_Toc190566186)

[Figure 28: Medical Dashboard Table 27](#_Toc190566187)

[Figure 29: Roster Table 27](#_Toc190566188)

[Figure 30: Scheduling System Table 27](#_Toc190566189)

[Figure 31: ER Diagram 28](#_Toc190566190)

# **Introduction**

The purpose of this design document is to provide a detailed overview of the key features and architectural design of the Elderly Care Management System(ECMS). The document provides information relating to the design of the application.

* The System Architecture Diagram that outlines the overall structure of the application outlining how the components interact with each other within the system.
* System Sequence Diagrams which illustrate the step-by-step flow of interactions between users and the system.
* GUI Prototype Screens have been designed to provide a visual representative of the applications interface.
* The Database Design to illustrate how the data is organized, stored, and managed within the system to ensure security.
* Class diagram to demonstrate the structure and illustrate the relationships between different classes of the system.

# **System Architecture**

A diagram of a computer server

AI-generated content may be incorrect.

Figure 1: System Architecture

This system architecture diagram illustrates the Elderly Care Management System(Care Net) a secure application which manages sensitive patient information. At its core, the Care Net application serves as the main interface for both Administrators and Carers, handling all patient data operations through a web interface. Patient data is secured through a dedicated Key Server that provides encryption and decryption via secure end points. The usage of tokens for authentication between the Webapp and the Key Server where all the data is then stored in the Care Net Database. The Key server has its own separate database for encryption keys and API tokens. This multi-tiered approach ensures that patient information remains secure while still being accessible to authorised users.

# **Technologies & Tools**

This project uses multiple “technologies and tools”. This section of the document outlines the technologies and tools used: Jupyter Notebook, Cryptography Library, Secrets Library, Maria DB, Python, Django, and FastAPI

**Jupyter Notebook**

Jupyter Notebook is utilized for prototyping and data exploration of the early phases of the project. This is particularly useful for testing database queries, experimenting with data encryption methods, and visualizing data.

**Cryptography Library**

The cryptography library is crucial to ensure data security within the application. This library was used to encrypt data and decrypt data providing robust protection for sensitive information handled by the system.

**Secrets Library**

The secrets library is integral for generating cryptographically strong random numbers used primarily for creating secure tokens within the system. The tokens generated were used to communicate between the Key Server and the Care Net app.

**MariaDB**

MariaDB was selected as the database management system to store and manage data throughout the development of this project. This was chosen as it has been reliable in previous projects. MariaDB stores all encrypted data for the ECMS(Care Net) application.

**Python**

Python will be the core language used in this project, helping with backend tasks, handling data, and connecting different parts of this system. It has many libraries that make it easy to work.

**Django**

Django is a high level python web framework. It will be used throughout the project, helping to enhance security and streamlining the development of the application.

**FastAPI**

FastAPI was used exclusively for developing the Key Server, FastAPI is a high-performance web framework that enables rapid API development. The Key Server developed using FastAPI acts as a stand-alone service, handling all encryption, decryption, token management and key management operations for the ECMS(Care Net) Application.

**Postman**

Postman is used during the development to interact with the Key Server API. Since the Key server currently lacks a UI. Postman allows for sending request to test API end points.

# **System Sequence Diagrams**

## **Core Features**

System Sequence Diagrams were created for the Core Features of the Application to illustrate the interaction between users and the system components. These diagrams provide a visual representation of the sequence of events that occur.

### **Key Server**

A diagram of a key server

AI-generated content may be incorrect.

Figure 2: Key Server Token and Key Generation System Sequence Diagram

This diagram illustrates the process of generating a new encryption key and token for secure communication between the Key server and the ECMS(Care Net) application. The Admin requests a new encryption key and token from the key server. The Key server generates these and stores them in the Key server database. The API token is displayed to the Admin who manually inputs it into the ECMS system. Once these credentials have been saved the ECMS can authenticate requests using the token, allowing for encryption and decryption via the Key server.

### **Patient Profile**

#### **Create Patient Profile**

A diagram of a diagram

AI-generated content may be incorrect.

Figure 3: Create Patient Profile System Sequence Diagram

This diagram illustrates the process of creating a new patient profile. The user initiates the creation of the patient profile by selecting the “Create Patient Profile” option. The system then displays a form allowing the user to enter required details.

Once the data has been entered the ECMS sends a request to the Key server to encrypt the sensitive data before storage. The Key server verifies authentication via an API token and then returns the encrypted data to ECMS. The ECMS (Care Net) stores the encrypted profile in the database, ensuring that patient information remains secure.

#### **Search Patient Profile**

A diagram of a software company

AI-generated content may be incorrect.

Figure 4: Search Patient Profile System Sequence Diagram

This diagram illustrates the process of searching for a patient profile. The user initiates the search by selecting the “Search Patient Profile” option. The system then displays search screen where the user enters the patient’s name or date of birth to locate the profile.

Once the search is initiated the ECMS queries the database to retrieve the encrypted patient profile. The encrypted profile is sent to the Key Server, where a request is made to decrypt. The Key Server authenticates the request using an API token before providing the decrypted patient profile back to ECMS, which then displays the profile to the user.

The diagram also has two alternative scenarios where a message is displayed if the patient profile is not found and where the token is incorrect and failed to decrypt the profile.

#### **Edit Patient Profile**

A diagram of a computer process

AI-generated content may be incorrect.

Figure 5: Edit Patient Profile Sequence Diagram

This diagram illustrates the process of editing patient profile after a patient search. The user initiates the edit by selecting the “Edit” button on the Patient Profile. The system then displays patient profile in editable mode where the user edits the patient information.

Once editing is completed the ECMS sends a request to the Key server to encrypt the profile with the new information. The Key server verifies authentication via an API token and then returns the encrypted data to ECMS. The ECMS stores the encrypted profile in the database.

The diagram also has an alternative scenario where a message is displayed if the patient profile failed to save.

#### **Delete Patient Profile**

A diagram of a diagram

AI-generated content may be incorrect.

Figure 6: Delete Patient Profile System Sequence Diagram

This diagram illustrates the process of deleting a patient profile after a patient search. The user initiates the delete by selecting the “Delete” button on the Patient Profile. The system then displays a confirmation to delete the patient profile.

Once the delete button has been clicked the ECMS sends a request to the Key server. The Key server verifies authentication via an API token and then allows for the deletion to continue . The ECMS then displays a message confirming deletion.

The diagram also has an alternative scenario where a message is displayed if the patient profile deletion failed.

### **Medical Dashboard**

#### **Search Patient Medical information on Dashboard**

A diagram of a software company

AI-generated content may be incorrect.

Figure 7: Search Patient Medical info on Dashboard System Sequence Diagram

This diagram illustrates the process of retrieving a patients medical information from the Medical Dashboard. The user initiates the process by selecting the “Medical Dashboard” . The system then displays the Medical Dashboard interface, where the user enters the patients name or date of birth.

Once the search is initiated the ECMS queries the database to retrieve the encrypted medical profile. The encrypted profile is sent to the Key Server, where a request is made to decrypt. The Key Server authenticates the request using an API token before providing the decrypted medical profile back to ECMS, which then displays the medical information of the patient to the user.

The diagram also has two alternative scenarios where a message is displayed if the medical profile was not found and where the token was incorrect and failed to decrypt the medical profile.

#### **Add Patient Medical information on Dashboard**

A diagram of a diagram

AI-generated content may be incorrect.

Figure 8: Add Patient Medical info on Dashboard System Sequence Diagram

This diagram illustrates the process of adding medical information of the patient via the Medical Dashboard. The user selects “Add” on the Patient Medical Dashboard. The system displays a form where the user can enter the new medical details for the patient.

Once the data is entered the ECMS sends a request to the Key server to encrypt the sensitive data before storage. The Key server verifies authentication via an API token and then returns the encrypted data to ECMS. The ECMS stores the encrypted profile medical information in the database.

#### **Edit Patient Medical information on Dashboard**

A diagram of a software company

AI-generated content may be incorrect.

Figure 9: Edit Patient Medical info on Dashboard System Sequence Diagram

This diagram illustrates the process of editing patient medical information after a patient search. The user initiates the edit by selecting the “Edit” button on the Profile. The system then displays medical information of the patient in editable mode where the user edits the patient information.

Once editing is completed the ECMS sends a request to the Key server to encrypt the profile with the new information. The Key server verifies authentication via an API token and then returns the encrypted data to ECMS. The ECMS stores the encrypted profile in the database.

The diagram also has an alternative scenario where a message is displayed if the patient medical information profile failed to save.

#### **Delete Patient Medical information on Dashboard**

A diagram of a process flow

AI-generated content may be incorrect.

Figure 10:Delete Patient Medical info on Dashboard System Sequence Diagram

This diagram illustrates the process of deleting patient medical information after a patient search. The user initiates the delete by selecting the “Delete” button on the Profile. The system then displays a confirmation to delete the medical information.

Once the delete button has been clicked the ECMS sends a request to the Key server. The Key server verifies authentication via an API token and then allows for the deletion to continue. The ECMS then displays a message confirming deletion.

The diagram also has an alternative scenario where a message is displayed if the deletion failed.

#### **Patient Medical Log Dashboard**

A diagram of a computer

AI-generated content may be incorrect.

Figure 11: Patient Medical Log Dashboard

This diagram illustrates the process of logging patient medical information after a patient search. The user initiates the log by selecting the “Log Medication/Details” button on the Profile. The system then displays a form where the user enters the necessary details such as which medication was administered, medications dosage when it was administered.

Once the data has entered the ECMS sends a request to the Key server to encrypt the sensitive data before storage. The Key server verifies authentication via an API token and then returns the encrypted data to ECMS. The ECMS stores the encrypted log in the database.

### **Roster**

A diagram of a schedule

AI-generated content may be incorrect.

Figure 12: Roster System Sequence Diagram

This diagrams illustrates the process of the roster system. The admin initiates to view the current schedule. The ECMS requests the scheduling system to fetch the current schedule.

The admin can then make any necessary modifications to the roster such as assigning carers or adjusting shifts. The updated schedule details are send to the Scheduling System, which checks availability and updates the database. Once confirmed it displays the updated schedule.

### **Care Planner**

#### **Add New Care Plan**

A diagram of a company

AI-generated content may be incorrect.

Figure 13: Add New Care Plan

This diagram illustrates the process of creating and managing care plans for patients within the ECMS system. The admin selects “Add” on the Care planner for the selected patient. The system displays a form where the user can enter the details for the care plan.

Once the data has entered the ECMS sends a request to the Key server to encrypt the sensitive data before storage. The Key server verifies authentication via an API token and then returns the encrypted data to ECMS. The ECMS stores the encrypted care plan in the database.

The carer then selects the care plan which is sent to the Key Server, where a request is made to decrypt. The Key Server authenticates the request using an API token before providing the decrypted care plan back to ECMS, which is then displayed to the carer who executes the assigned tasks and marks them as complete.

#### **Delete Care Plan**

A diagram of a diagram

AI-generated content may be incorrect.

Figure 14: Delete Care Plan

This diagram illustrates the process of deleting a care plan of a selected patient. The admin initiates the delete by selecting the “Delete” button on the Care planner. The system then displays a confirmation to delete the care plan.

Once the delete button has been clicked the ECMS sends a request to the Key server. The Key server verifies authentication via an API token and then allows for the deletion to continue. The ECMS then displays a message confirming deletion.

The diagram also has an alternative scenario where a message is displayed if the deletion failed.

## **Non Core Features**

Following are the Non core features identified which may be implemented in this projects timeline:

**Login & Register:** Secure login functionality which will allow users to access the system, while admins will have the ability to register new carers.

**Incident Reporting:** Carers will be able to report incidents that occur during patient visits, such as falls or injuries. These reports will include detailed account of the event and what course of action was taken.

**Alerts:** Admin and Carers will receive notifications. For example: Admins will be alerted of a new incident report, while Carers will receive reminders to complete tasks such as administrating medication to a patient.

# **Class Diagram**

A diagram of a computer program

AI-generated content may be incorrect.

Figure 15: Class Diagram

# **Prototype GUI Screens**

## **Main Menu**

A blue background with orange labels

Description automatically generated

Figure 16: Main Menu

## **Patient Profile Prototype**

A screenshot of a contact form

AI-generated content may be incorrect.

Figure 17: Patient Profile

## **Medical Dashboard**

A screenshot of a medical dashboard

AI-generated content may be incorrect.

Figure 18: Medical Dashboard

## **Medical Dashboard Log**

A screenshot of a medical dashboard

AI-generated content may be incorrect.

Figure 19: Medical Dashboard Log

## **Roster**

A screenshot of a calendar

AI-generated content may be incorrect.

Figure 20: Roster

## **Care Planner Admin**

A screenshot of a computer

AI-generated content may be incorrect.

Figure 21: Care Planner Admin

## **Care Planner Carer**

A screenshot of a computer

AI-generated content may be incorrect.

Figure 22: Care Planner Carer

# **Database Design**

## **Key Server Database**

Key Server is a sperate application which has its own database and is responsible for securely handling encryption, decryption and authentication. This database stores only the encryption key and the API tokens.

### **Key Management Table**

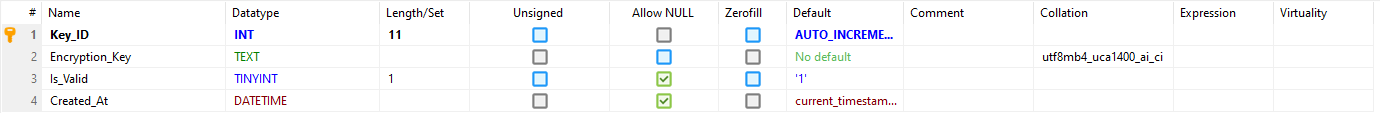


Figure 23: Key Management Table

### **Key Server Api Token Table**

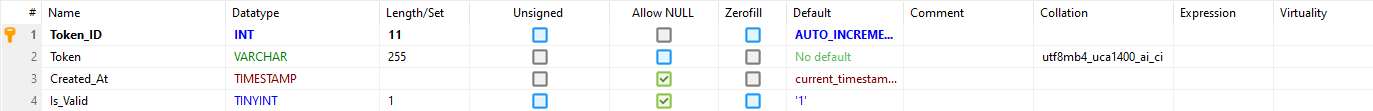


Figure 24: API Token Table

## **Care Net Database**

Care Net Database serves as the primary data storage system for the Elderly Care Management System. It is responsible for securely handling and managing storage of all encrypted data from Patient Profile, Care Planner, Roster and Medical Dashboard. The database structures shown below is subject to change as the system changed to meet requirements.

### **Patient Profile Table**

A screenshot of a computer screen

Description automatically generated

Figure 25: Patient Profile Table

### **Users Table**

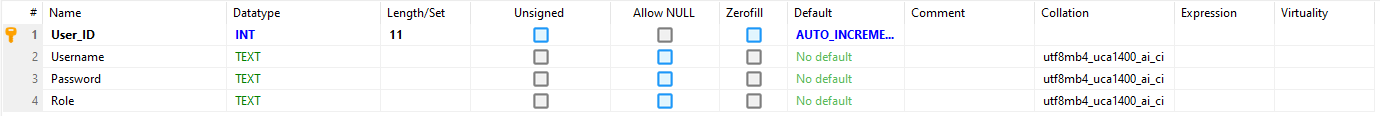


Figure 26: Users Table

### **Care Planner Table**

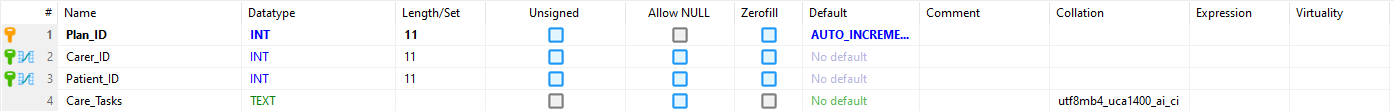


Figure 27: Care Planner Table

### **Medical Dashboard Table**

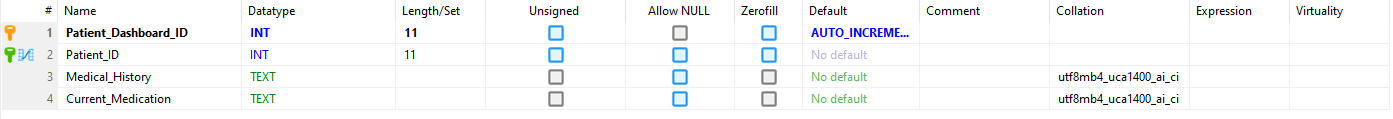


Figure 28: Medical Dashboard Table

### **Roster Table**

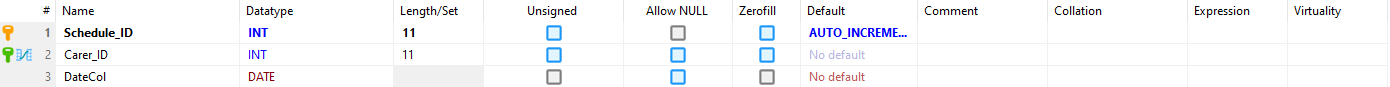


Figure 29: Roster Table

### **Scheduling System Table**

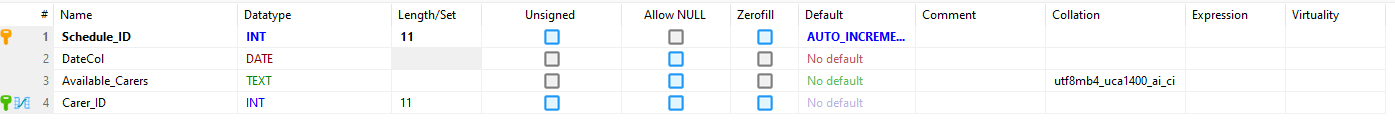


Figure 30: Scheduling System Table

# **ER Diagram**

A computer screen shot of a diagram

AI-generated content may be incorrect.

Figure 31: ER Diagram

# **Conclusion**

This document has successfully outlined the design features of the Elderly Care Management Application(Care Net). It provides a comprehensive overview of the tools, technologies utilised in the development process. System Sequence Diagrams and class diagrams have been included to visually represent the interactions within the application. Additionally, the database structure demonstrates how the data is stored. Prototype Screens showcasing the initial view of the app have also been included.

# **References**

Django Software Foundation, 2024. *Django.* [Online]   
Available at: https://www.djangoproject.com/  
[Accessed December 2024].

FastAPI Tiangolo, 2025. *FastAPI.* [Online]   
Available at: https://fastapi.tiangolo.com/  
[Accessed 2025].

Jupyter, 2024. *Jupyter.* [Online]   
Available at: https://jupyter.org/  
[Accessed December 2024].

MariaDB Foundation , 2024. *MariaDB Server.* [Online]   
Available at: https://mariadb.org/  
[Accessed December 2024].

PYPI, 2024. *Cryptography.* [Online]   
Available at: https://pypi.org/project/cryptography/  
[Accessed December 2024].

Python Software Foundation, 2024. *Python.* [Online]   
Available at: https://www.python.org/  
[Accessed December 2024].

Python Software Foundation, 2025. *Secrets Library.* [Online]   
Available at: https://docs.python.org/3/library/secrets.html  
[Accessed 2025].