

**TEAM 12 FEVER**

**ARCHITECTURE DESIGN**

22/05/2022

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Author | Description |
| 25/05/2022 | 1.0 | Lê Hữu Hiệp | Tạo template |
| 26/03/2022 | 1.0 | Lê Hữu Hiệp | Vẽ phần 2.1 Context Diagram |
| 08/06/2022 | 1.2 | Bùi Công Danh | Vẽ View Physical |
| 08/06/2022 | 1.2 | Bùi Công Danh | Viết rationel cho View physical |
| 08/06/2022 | 1.2 | Bùi Công Danh | Viết Element Responsibilities cho View physical |
| 08/06/2022 | 1.2 | Lê Hữu Hiệp | Rà soát lại tài liệu |
| 08/08/2022 | 2.0 | Lê Hữu Hiệp | Duyệt tài liệu |

Contents

[Table of Figures 3](#_Toc446424501)

[Table of Responsibility Catalogs 4](#_Toc1229249329)

[1 Introduction 4](#_Toc2008491786)

[1.1 About this document 5](#_Toc1299505497)

[2 Architectural Documentation 5](#_Toc1775430736)

[2.1 Context Diagram 5](#_Toc676569030)

[2.2 Static Perspective 5](#_Toc85991994)

[2.3 View personal information Level 1 8](#_Toc928535651)

[2.4.3 Relationship Resonsibilities Cataog 12](#_Toc582801102)

[2.5.1 Decomposition Retional 13](#_Toc707785197)

[2.5.2 Element Responsibilities Catalog 13](#_Toc1549778298)

[2.5.3 Relationship Resonsibilities Cataog 14](#_Toc103428925)

[2.5.4 Behavior 14](#_Toc2133878679)

[2.3 Physical Perspective 15](#_Toc2111396479)

[3 Mappings between Perspectives 17](#_Toc773415259)

[3.1 Mapping between Static and Dynamic 17](#_Toc1617501113)

[3.2 Mapping between Dynamic and Physical 17](#_Toc963520689)

# Table of Figures

￼

￼

￼

￼

￼

￼

￼

￼

# Table of Responsibility Catalogs

￼

￼

￼

￼

￼

￼

￼

￼

￼

￼

￼

￼

￼

￼

￼

# 1 Introduction

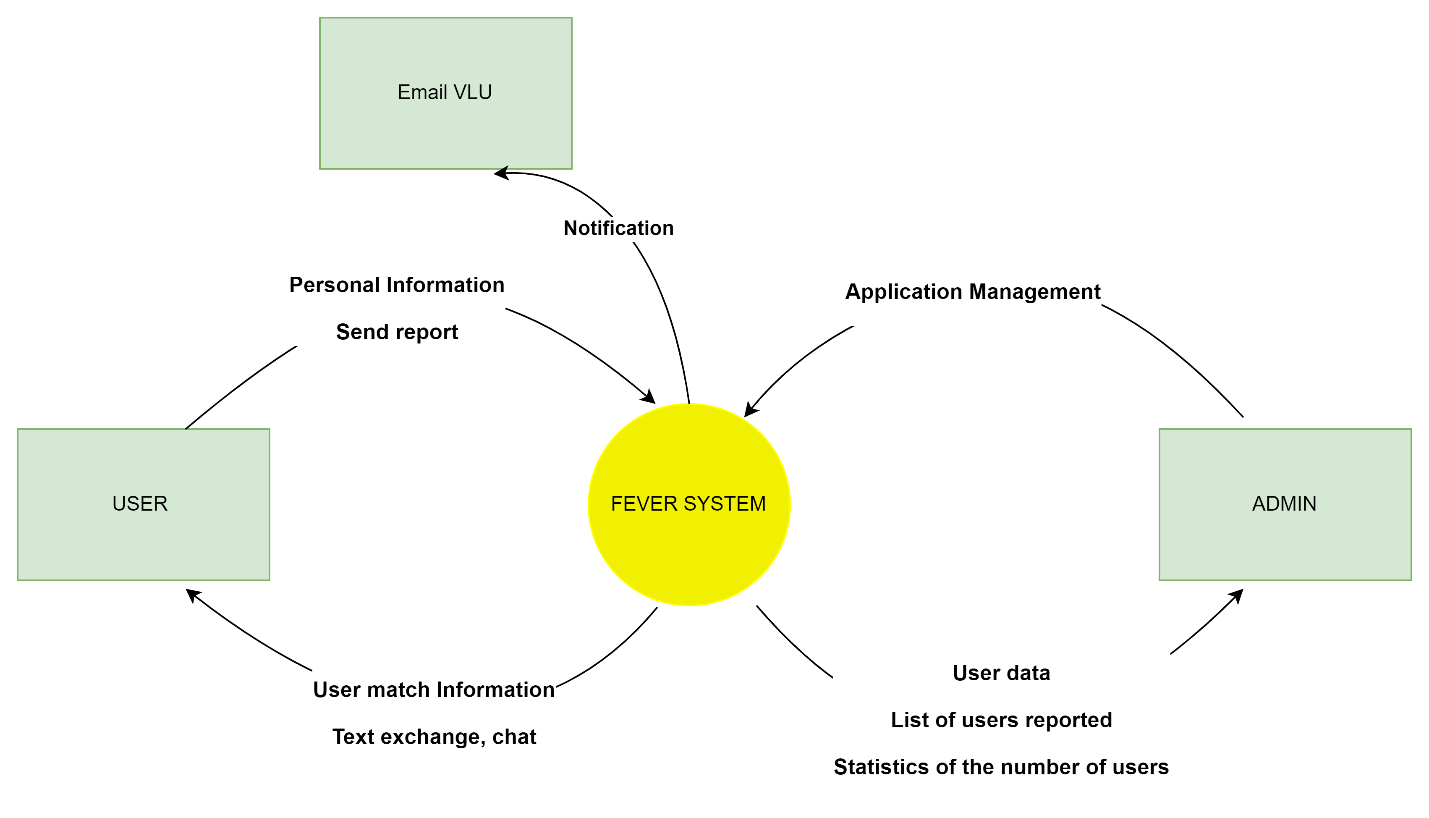
## About this document

The purpose of this document is to describe and design the architecture for the dating application project.

# Architectural Documentation

This section contains architecture diagrams and associated responsibility lists. Except for the context diagram and the collaboration diagram, every other subsection represents a level of separation. Each decomposition level diagram is accompanied by a separation rationale, an element liability category, and a relationship liability category.

## Context Diagram



## Static Perspective

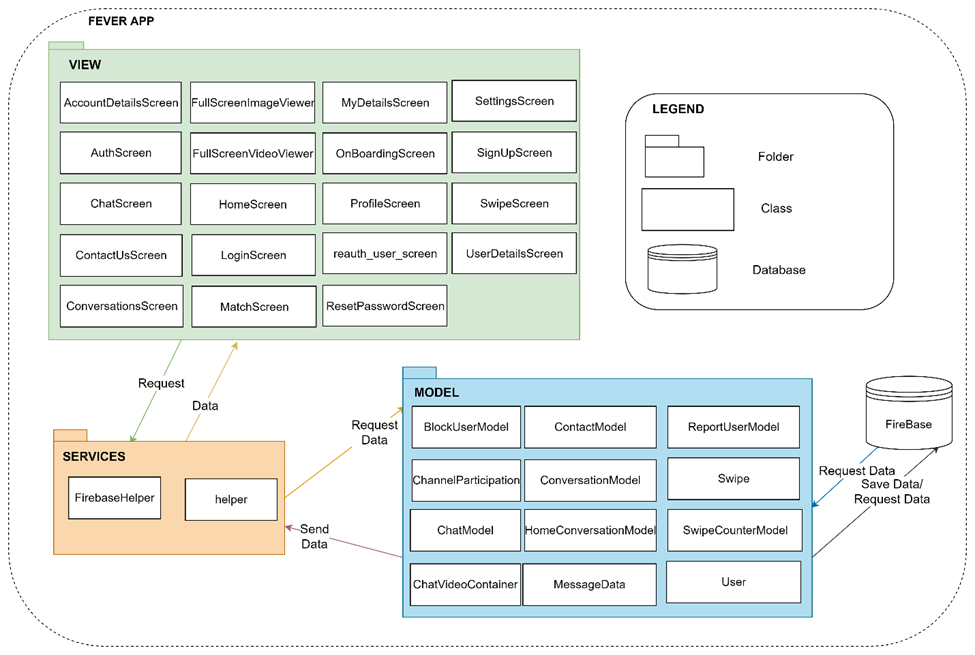


Figure 2 Sơ đồ Static Perspective

#### 2.2.1 Perspective Rationale

|  |  |
| --- | --- |
| **Associated Drawings: Figure 2**  **Associated Responsibilities:** Catalog 1, Catalog 2 | **Perspective: Static** |
| In this section, the diagram shows the components of the system. These components interact with each other to process data, the components in the system all have separate tasks | |

#### 2.2.2 Element Responsibilities Catalog

|  |  |
| --- | --- |
| **Associated Drawings: Figure 2** | **Perspective: Static** |
| **Element** | **Responsibilities** |
| Services | The execution process handles requests from the user, retrieves the required data from the Model, and returns the appropriate data. |
| View | Contains the user interface, gets data from Services and displays the data to the user. |
| Model | Contains Classes that connect to Models in the data processing database. |
| Database | Store data and send data |

#### 2.2.3 Relationship Responsibilities Catalog

|  |  |
| --- | --- |
| **Associated Drawings: Figure 2** | **Perspective: Static** |
| **Relationship** | **Responsibilities** |
| View request to Service | When the user proceeds to access the app, the view will now send a request to the Service |
| Service send data to View | After the request is sent to the Service, the Service will now send the appropriate data back to the request |
| Service request data to Model | Service sends request to Model to request data accordingly |
| Model send data to Service | Receive information and return data results on demand |
| Model request data to Firebase | Model sends request to firebase to get requested data or save new data to firebase |
| Firebase send data to Model | Store data and return data on demand |

### 2.3 View personal information Level 1

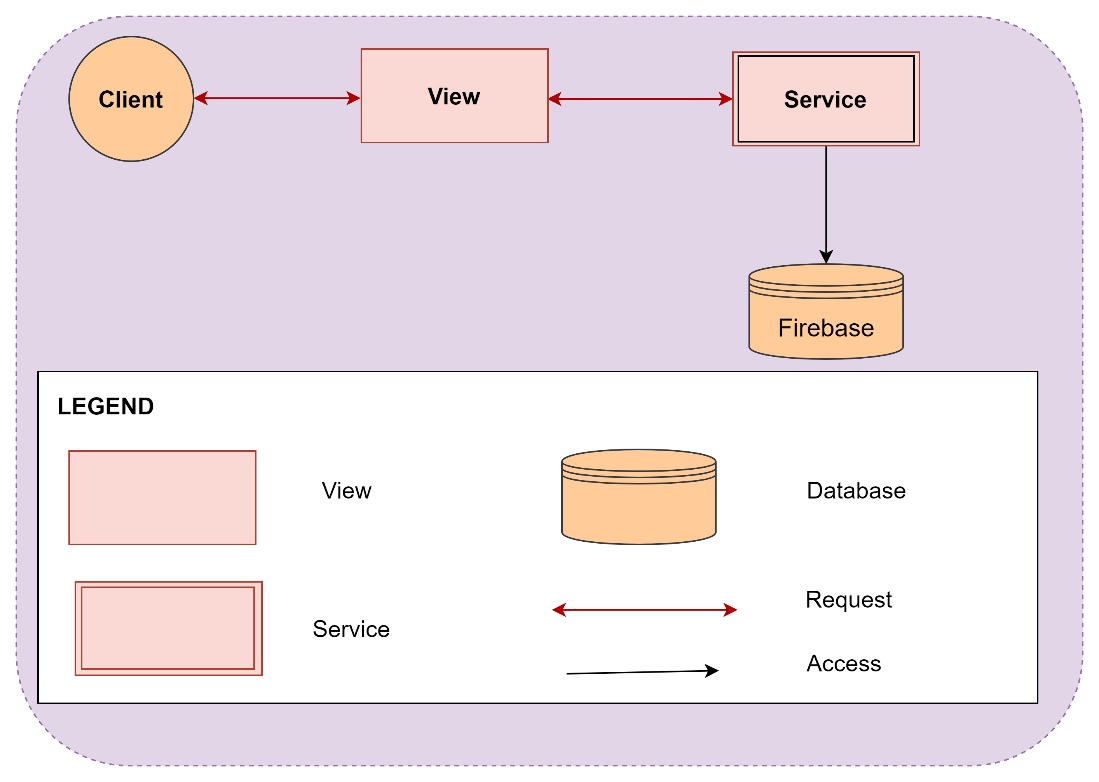


Figure 3

#### 2.3.1 Perspective Rationale

|  |  |
| --- | --- |
| **Associated Drawings:** Figure 3  **Associated Responsibilities:** Catalog 3, Catalog 4 | **Perspective:** Dynamic |
| This design is used to satisfy the following quality attributes:  • QAS01 (Performance): User Profile Viewing Performance  This is the part that shows the data flow of the system and shows the user's interaction with the system. Data warehouse to store all the information related to the user's request.  Figure 3 above shows the split layer of viewing personal information. When users log in to the app, users can view personal information, see the list of compatible people in the fastest way and do not have a bottleneck in the database due to too many requests. | |

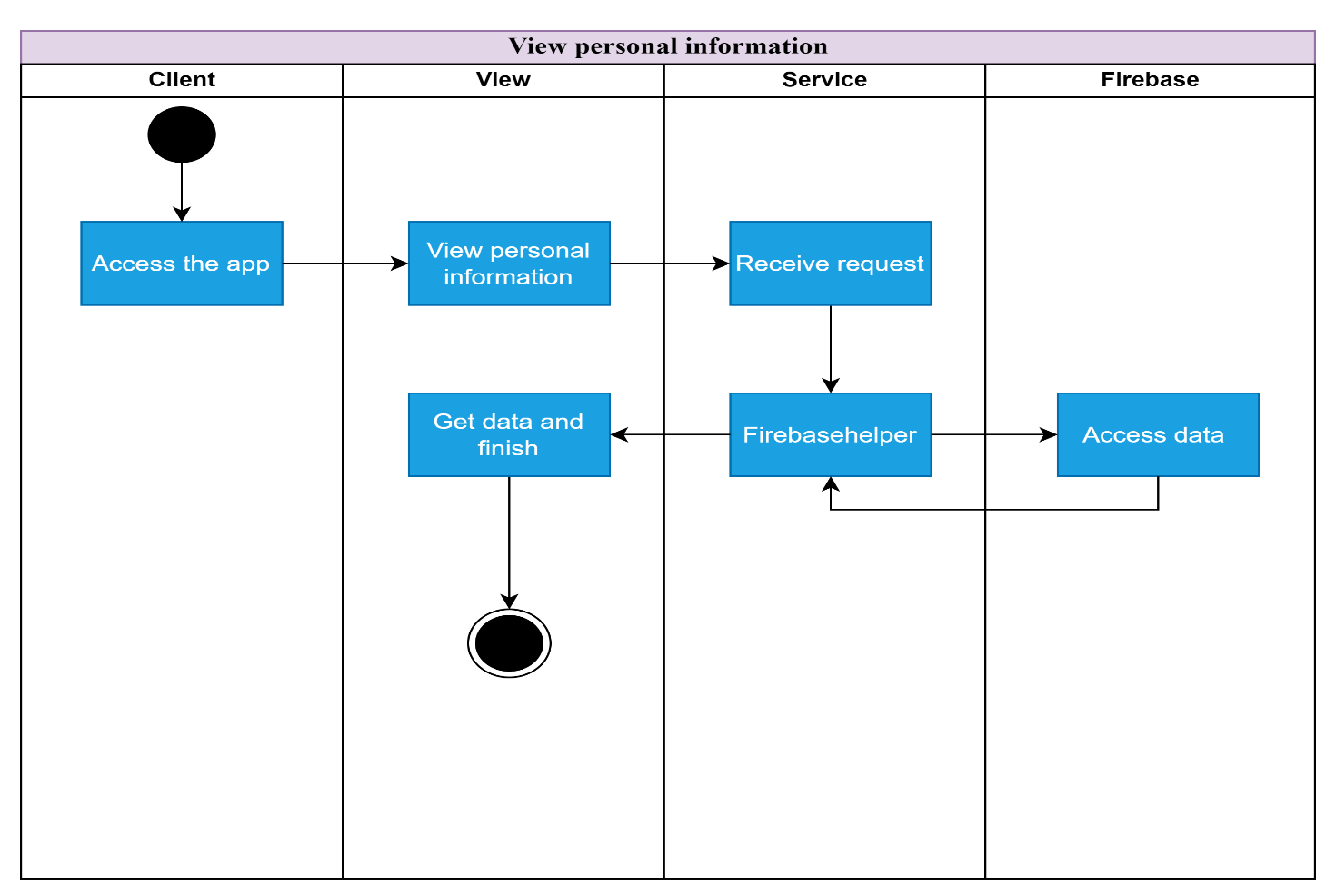
#### 2.3.2 Element Responsibilities Catalog

|  |  |
| --- | --- |
| Associated Drawings: Figure 3 | Perspective: Component-and-Connector |
| Element | **Responsibilities** |
| Users | Can interact with the system  Accessible to the app |
| View | The interface of the app Fever |
| Service | Receiving and processing information |
| Database | Save data |

#### 2.3.3 Relationship Responsibilities Catalog

|  |  |
| --- | --- |
| Associated Drawings: Firgure 3 | Perspective: Component-and-Connector |
| Relationship | **Responsibilities** |
| User requests to View | User sends an access request to the application interface |
| View request to Service | View receives request from user and then sends to Service |
| Service request to Database | The database receives the request from the Service and responds with data according to the request |

2.3.4 Behavioral



#### 2.4 Fastload view user profiles Level 1

Diagram

Description automatically generated

**2.4.1 Decomposition Retional**

|  |  |
| --- | --- |
| **Associated Drawings:** Figure 4  **Associated Responsibilities:** | **Perspective:** Dynamic |
| This is the part that shows the data flow of the system and shows the user's interaction with the system. Data warehouse to store all the information related to the user's request.  When the user accesses the user profile page, the system will load the page within 2 seconds  This architectural design aims to be able to meet the needs of users, while improving the system's performance to ensure the fastest and most accurate system. | |

**2.4.2 Element Responsibilities Catalog**

|  |  |
| --- | --- |
| **Associated Drawings: Figure 4** | **Perspective: Component-and-Connector** |
| **Element** | **Responsibilities** |
| **Users** | * Can interact with the system * Have access to the app |
| **Mydetailscreen** | The interface of the view user profile |
| **Service** | Receiving and processing information |
| **Database** | Save data |

### 2.4.3 Relationship Resonsibilities Cataog

|  |  |
| --- | --- |
| Associated Drawings: Firgure 4 | Perspective: Component-and-Connector |
| Relationship | **Responsibilities** |
| User requests to View | User sends an access request to the application interface |
| View request to Service | View receives request from user and then sends to Service |
| Service request to Database | The database receives the request from the Service and responds with data according to the request |

#### 2.5 FastLoad view user profiles Level 2:

Diagram

Description automatically generated

### 2.5.1 Decomposition Retional

|  |  |
| --- | --- |
| **Associated Drawings:** Figure 5  **Associated Responsibilities:** Catalog 9, Catalog 10 | **Perspective:** Dynamic |
| At this level, we will go deeper into the decomposition of the Service. After the decomposition, we will see more clearly about the information processing of the data  Cache\_Image will receive requests from outside and then send them to Index and Firebasehelper at the same time. Save Data To Memory will save it to the Cache Dictory to store it again, later if that information is found again, it will find it quickly  Firebasehelper will receive information from Cache\_Image and return it to the View if the data is found for the first time. If that data is retrieved again, Firebasehelper will retrieve it from the previously stored Cache Dictory. | |

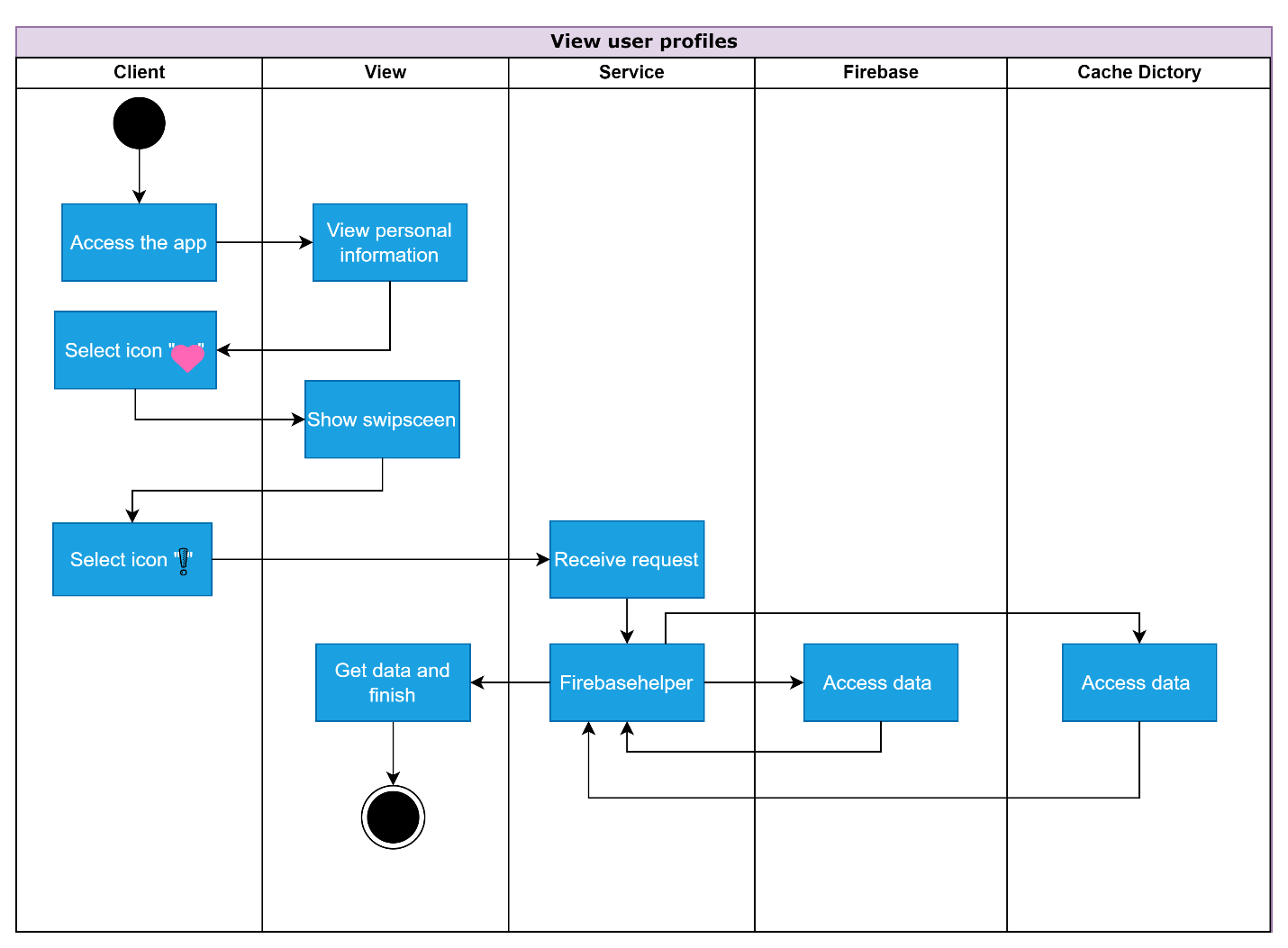
### 2.5.2 Element Responsibilities Catalog

|  |  |
| --- | --- |
| **Associated Drawings: Figure 7** | **Perspective: Component-and-Connector** |
| **Element** | **Responsibilities** |
| **Cache\_Image** | Get pictures from the outside in |
| **Save Data To Memory** | Store recently accessed data in the clipboard |
| **Firebasehelper** | Return the found results to View |

### 2.5.3 Relationship Resonsibilities Cataog

|  |  |
| --- | --- |
| **Associated Drawings: Firgure 7** | **Perspective: Component-and-Connector** |
| **Relationship** | **Responsibilities** |
| **Cache\_Image respond to Save Data** | After receiving the request from the outside, Save Data To Memory to store the data again |
| **Save Data To Memory Access Data đến Cache Dictory** | Save Data To Memory will access in Cache Dictory and perform data storage again |
| **Receive Data respond to Response Data** | In addition to sending it to Save Data, Cache\_image does one more thing, it is sent to Firebasehelper to quickly return to the user. |
| **Response Data get data from Data Dictory** | Access to Data Dictory to get previously requested data. Reduce the request time to the Server and return the results quickly to the user |

### 2.5.4 Behavior



## Physical Perspective

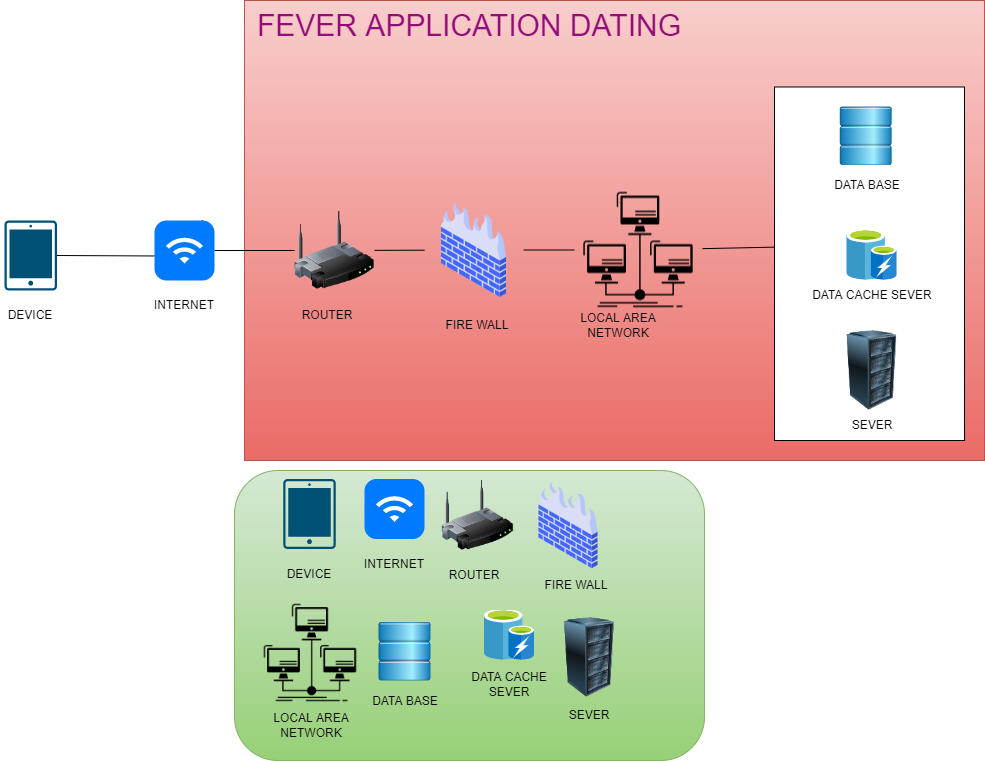


Image 8

#### Perspective Rationale

|  |  |
| --- | --- |
| **Associated Drawings:** Image 8  **Associated Responsibilities:** Catalog 15 | **Perspective:** Physical |
| Figure 8 above shows the Physical Perspective design of the system. The design represents the components (Hardwares and Software) that need to be used for the system to satisfy the quality attributes in this architectural document. | |

#### Element Responsibilities Catalog

Catalog 15 – Elements – Physical Perspective

|  |  |
| --- | --- |
| **Associated Drawings:** Image 8 | **Perspective:** Physical |
| **Element** | **Responsibilities** |
| Device | This element is the personal and organizational devices used to access the system |
| Internet | This element is used to connect the customer's machine to the system and is a tool to help the application connect with the user. |
| Router | This element is a router, a network device that forwards data packets between computer networks |
| Firewall | This element is a firewall, which is a network security system, which acts as a barrier between safe and unsafe networks, controlling incoming and outgoing network traffic based on predefined rules. |
| Local Area Network | Element is a computer network that connects computers in a limited area |
| Data server | This element is the system's Database used to store lottery results. The application will use PostgreSQL - an object and relational database management system. |
| Data Cache Server | This element is used to configure Redis Cache to read data from PostgreSQL and save Cache data into the user's application, thereby helping to improve the performance of the application, avoiding too many data queries directly into the database as a database. overloaded. |
| Server | This element is a computer connected to a computer network or the Internet, with a static IP, with high processing power. On its people install software to serve other computers to access to request services and resources. |

# Mappings between Perspectives

## Mapping between Static and Dynamic

|  |  |  |
| --- | --- | --- |
| Modules | Component | Comments |
| Services | * Receive data * Process data |  |

## Mapping between Dynamic and Physical

|  |  |  |
| --- | --- | --- |
| Component | Element | Comments |
| Firebase | Data Server |  |
| Cache - directory | Data Cache Server |  |