|  |  |
| --- | --- |

**eProject Document**

**Health Insurance System**

| **Group** | |
| --- | --- |
| **Members** | VU NGOC ANH TU – Student1290590(Team Leader)  VI NGOC LONG - Student1346407  TRAN HUY VIET - Student1341338  PHAM HOANG TU - Student1370671 |
| **Instructor** | Bui Thanh Hai |
| **Batch** | C2110I2 |
| **Semester** | III |

**🙠 Issued Jan-02-2024 🙢**

**Table of Content**

1. **Problem definition**
   1. ***Introduction***
   2. ***Existing problems***
   3. ***Proposed solution***
2. **Application functions**
3. **Architecture and Design of the System**
4. **Flow Chart**
5. **Data Flow Diagram**
6. **Database Design/Structure**
   1. ***Database relationship diagram***
   2. ***Table design***
7. **Some typical interface**
8. **Check List of Validation**

1. **Problem Definition**
   1. **Introduction**

Health Insurance System is a program to maintain all information related to the employees (Patients), information about insurance companies and medicals to the corporate employees. This system can also maintain the health related and all information of the employees.

* 1. **Existing problems**
* It is limited to a single system.
* It is less user-friendly.
* It is having lots of manual work (Manual system does not mean that you are working with pen and paper, it also include working on spreadsheets and other simple softwares
* It requires more numbers of employees need to work.
* It is a time consuming process.
* The present system is very less secure.
* It is unable to Maintain Employee specific information and also their policy info.
  1. **Proposed solution**
* All users must have an account provided by the admin to login to be able to see their information. There will be an authentication process to verify identity between the employee and the administrator.
* The program is highly secured and requires authenticated users and their respective role to access information
* A lot more user friendly UIs
* Find and assign policy with ease and as accurately as possible.
* Requires a lot less employees to work

**The program should enable the members to do the following:**

### Policy:

* Being able to request a new policy with given insurance companies attached. This request will be sent to the admin, waiting for approval.
* Ordering with the given policies approved by the admin. Then that policy will be added to their profile.

**Profile details:**

* Employee will be able to see all their informations and attached policy.
* If the employee forgot their password, there will be an option to change their password.

**The admin will have all the privileges to control, create new accounts and insurance companies, monitor policies and transactions made by employees.**

1. **Application functions**

| **User** | **Functions** |
| --- | --- |
| **Guest** | * Register * Login * Change passwords |
| **Member** | * View all personal details * View current attached policy * Create new policies * Ordering policies * Change passwords |
| **Admin** | * Log in * Employee Account: only the admin can create a member account, additionally edit, delete and add policy to that account. The admin can also search for members by their full name for convenience. * Company: can view all details, create, edit and delete insurance companies. * Policy: can view all details, create, edit and delete. * Policy Request: Can approve or deny the requests made by employees. * Policy status: see details and status of requested policies. * Bill: see details of transactions made by employees, in this case ordering their policy. * Change password |

* 1. **Common functions**

| **Insert** | |
| --- | --- |
| Input | Information to be added |
| Process | Enter information into the system. New add execution system |
| Output | New information added |

| **Update** | |
| --- | --- |
| Input | Information that needs to be changed |
| Process | Search for the item to be corrected, modify the information and update the changes |
| Output | Information is changed |

| **Delete** | |
| --- | --- |
| Input | Information to be deleted |
| Process | Search for the item to be deleted, delete the item from the system |
| Output | Delete information from the database |

| **Search** | |
| --- | --- |
| Input | Parameters |
| Process | System for searching information by first name and last name |
| Output | Return lists of objects |

| **Login** | |
| --- | --- |
| Input | Fill in the information fields |
| Process | Check for correct username and password |
| Output | Return Jwt token for authentication |

* 1. **Admin functions**

| **Approve Policy** | |
| --- | --- |
| Input | Selected policy for action |
| Process | If approved, start generate new policy with the existing information  if denied, policy will be rejected |
| Output | New policy added |

* 1. **Member functions**

| **Order** | |
| --- | --- |
| Input | Select the available policies |
| Process | Create new policy profile to that user, generate transaction bills to the admin |
| Output | Order Success |

| **Login** | |
| --- | --- |
| Input | Fill in the information fields |
| Process | Check for correct username and password |
| Output | Jwt token |

1. **Architecture and design of the system**

The Application is made of three-tier architecture.



User interface





Server handling logic, retrieve and manipulate

data

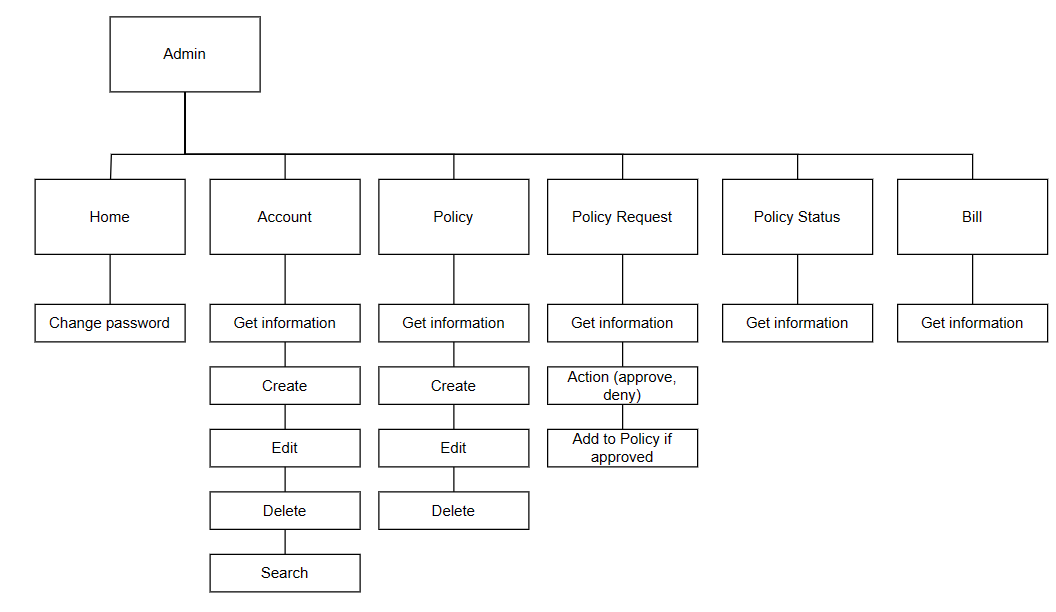


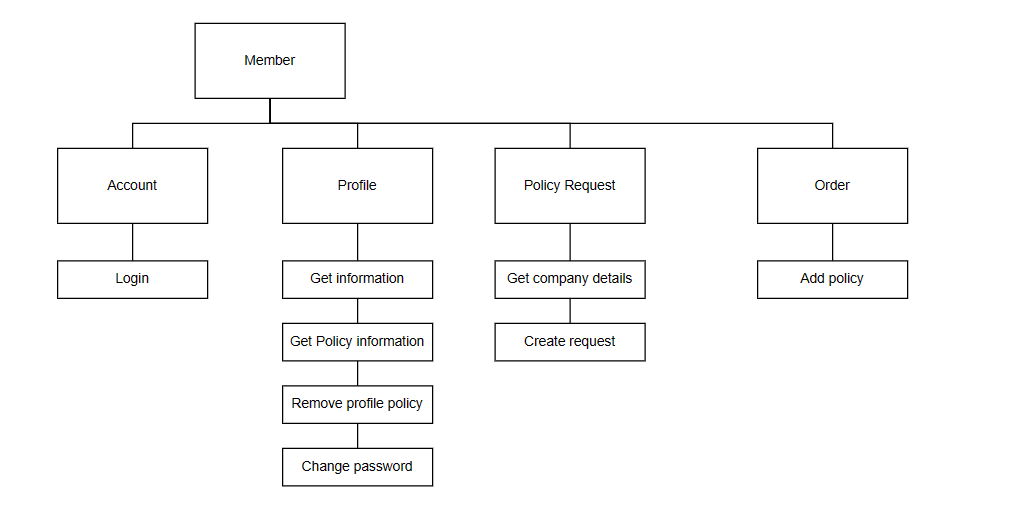


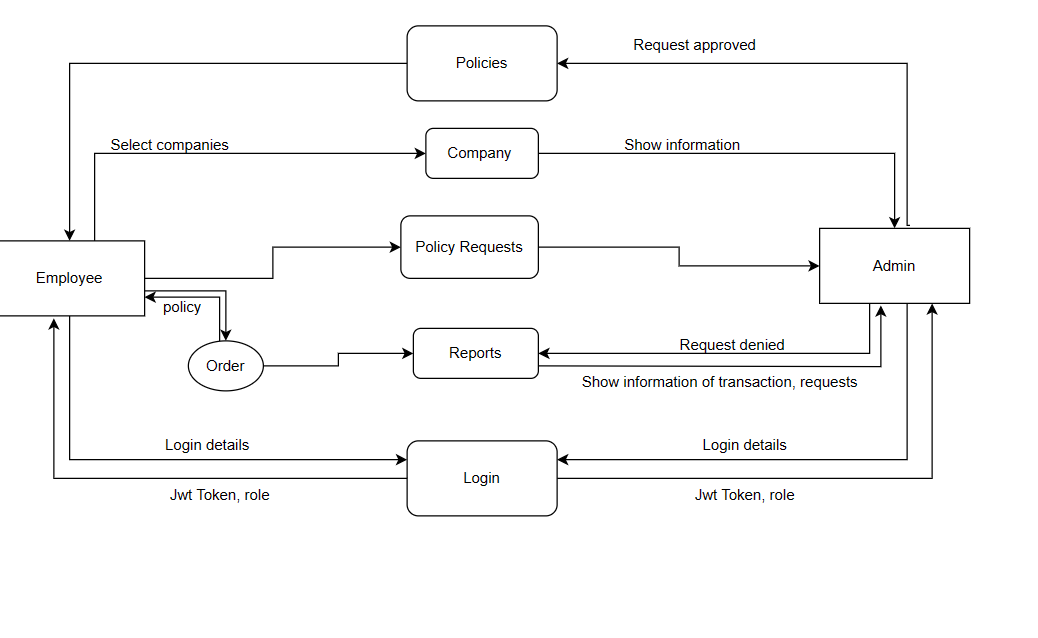
Store data

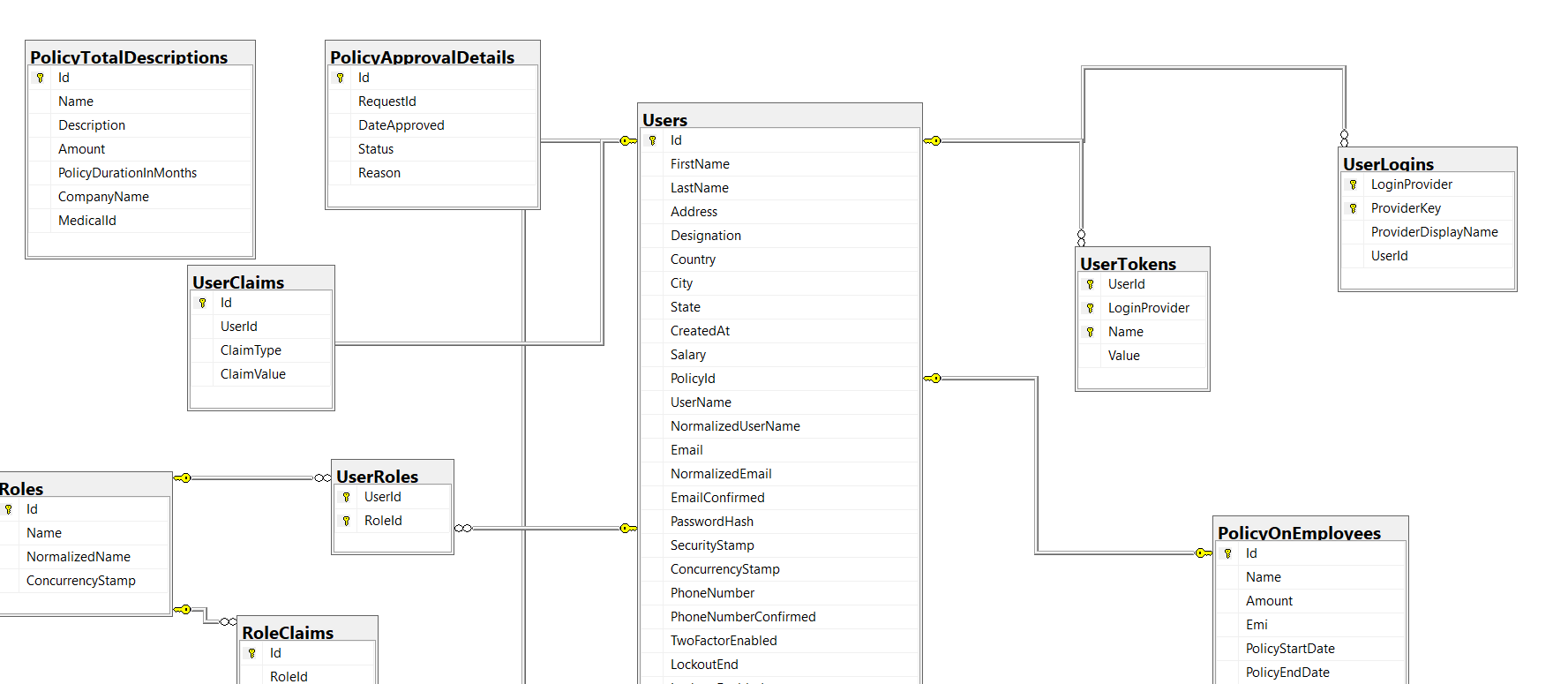
**Web – based distributed 3 - Tier Architecture of the Project**

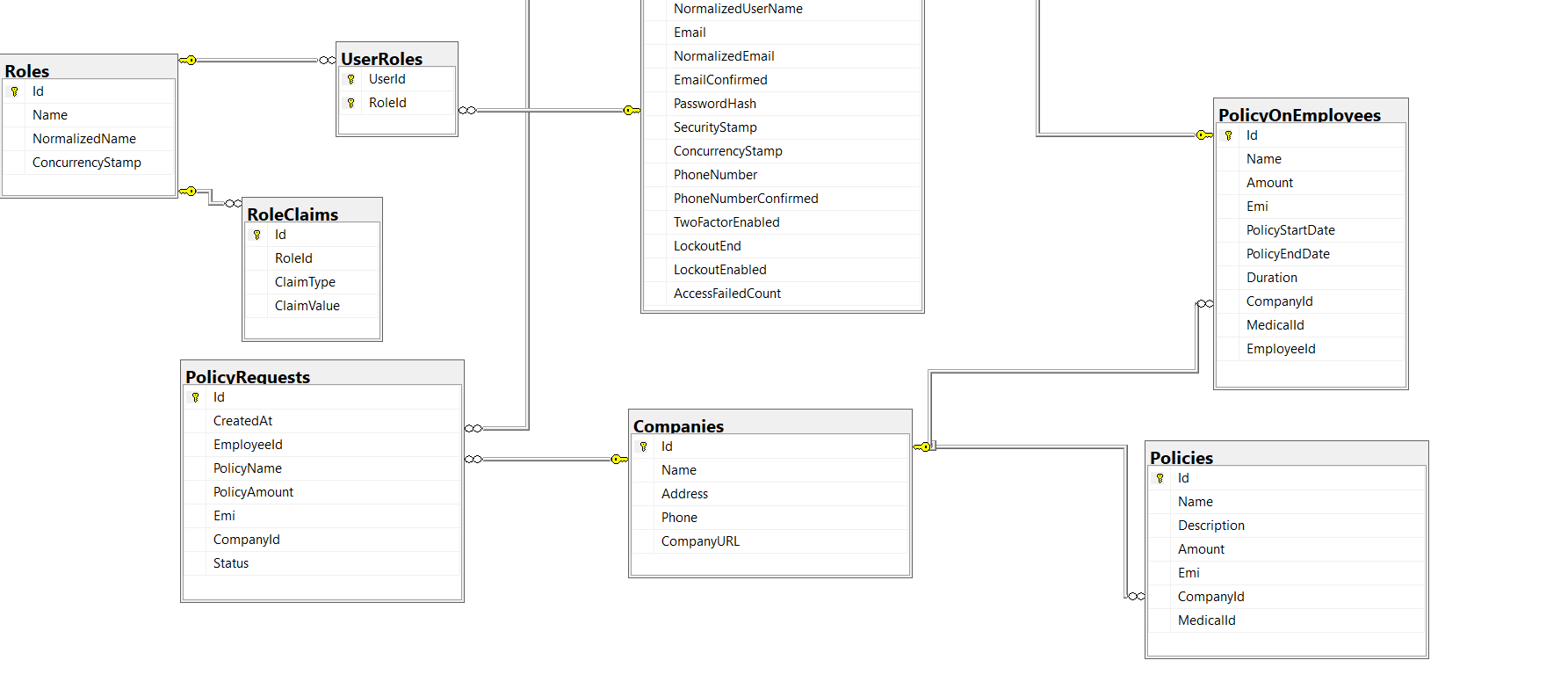
1. **Flow chart**

****

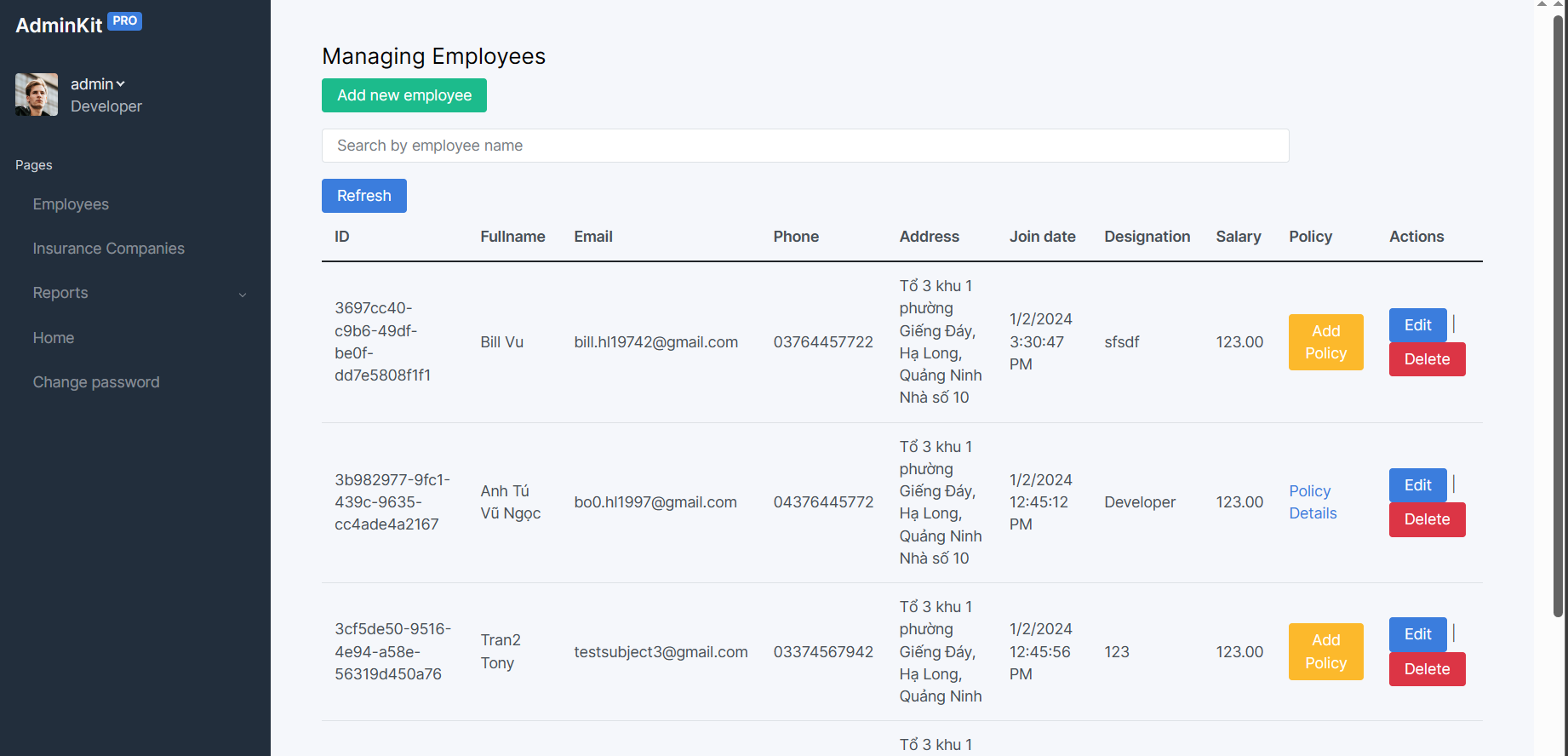
****

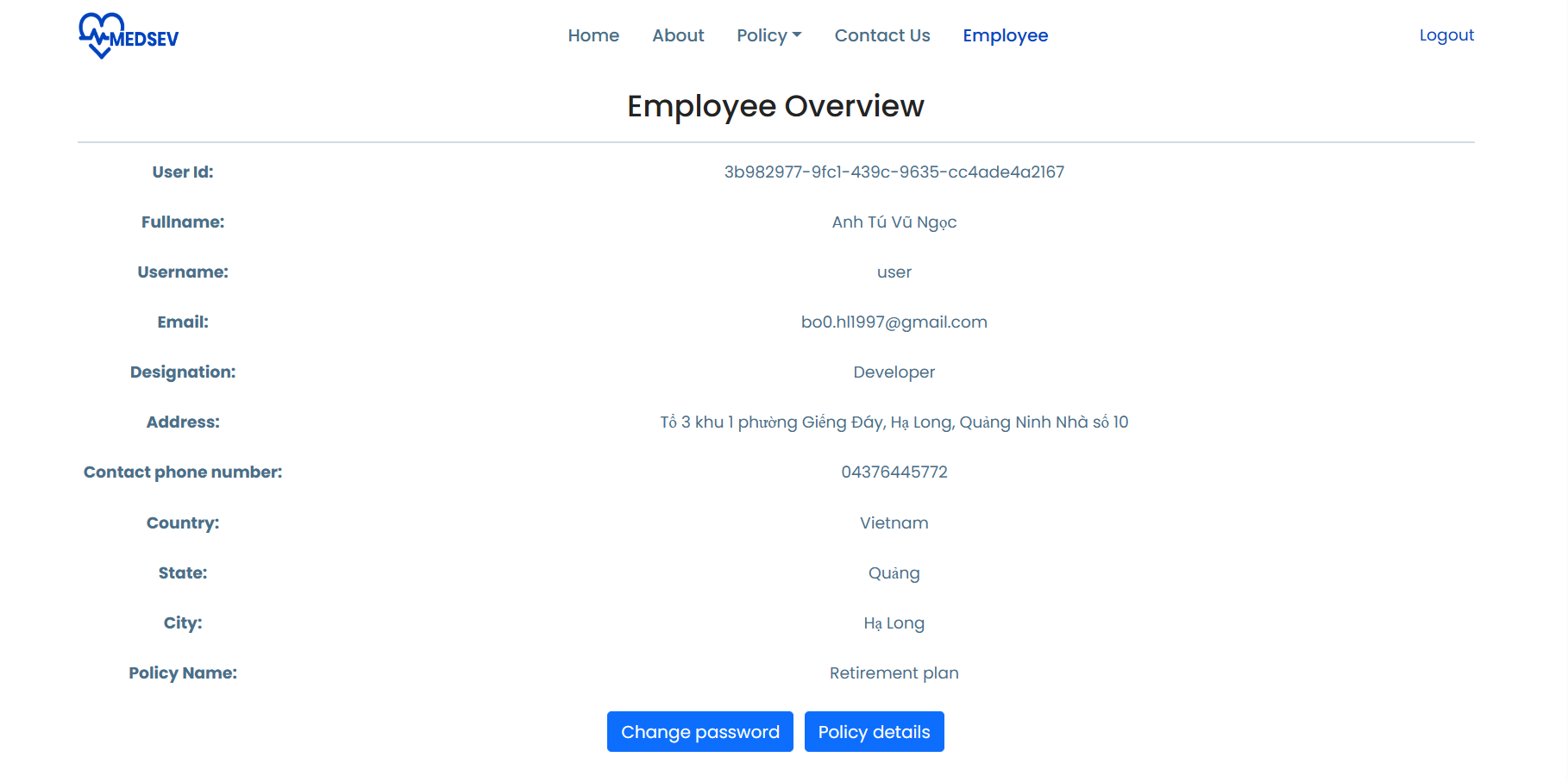
1. **Data flow diagram**
   1. **General Diagram** **
2. **Database Design/Structure**

****

**

1. **Some typical interface**



**

1. **Check list of validation**

| Option | Validate |
| --- | --- |
| Insert : Account | Yes |
| Edit : Account | Yes |
| Login : can enter either username or email address | Yes |
| Register | Yes |

**THE END**