# National Autonomous University of Mexico

School of Engineering

**Object-oriented programming models**

Software Design

Version :1.0

9 of november 2021

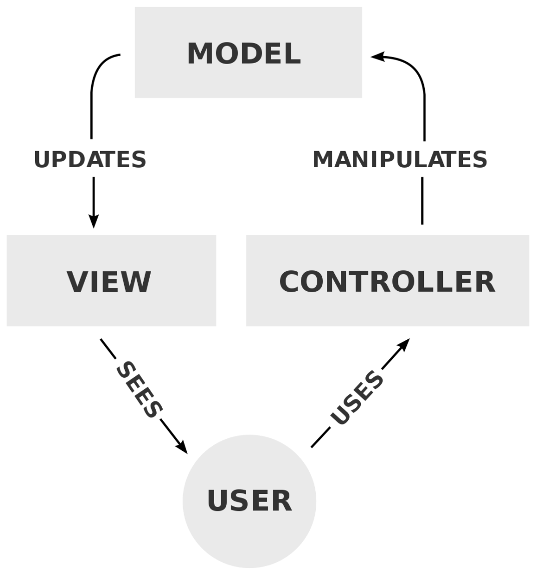
## Table of content:

## Changelog

| Version | Description | Responsible for the update | Update date |
|---|---|---|---|
| v1.0 | Document creation | José Alejandro Morán Duque | 7/10/2021 |
| v2.0 | Added updated diagrams | José Alejandro Morán Duque | 2/12/2021 |
| | | | |

# 1. Architecture description

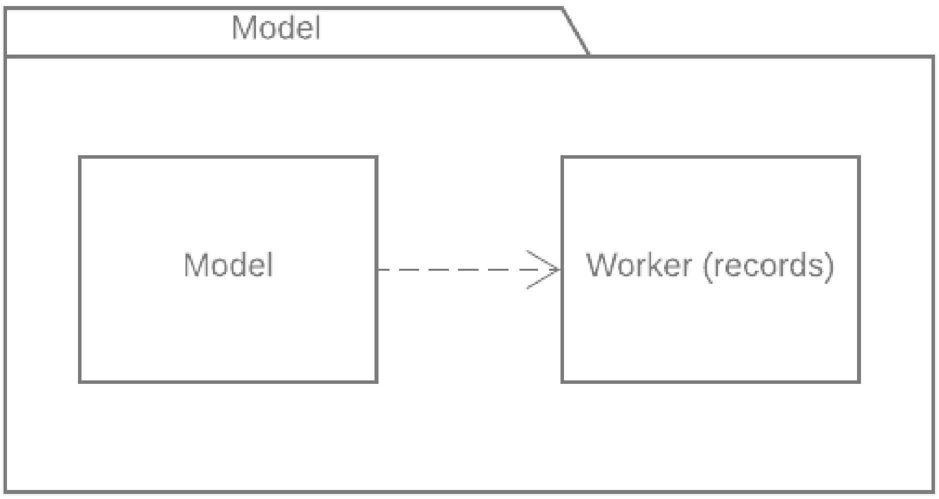
The used software architecture will be MVC or Model View Controller and it will be implemented in java



## 1.1. Architecture packages

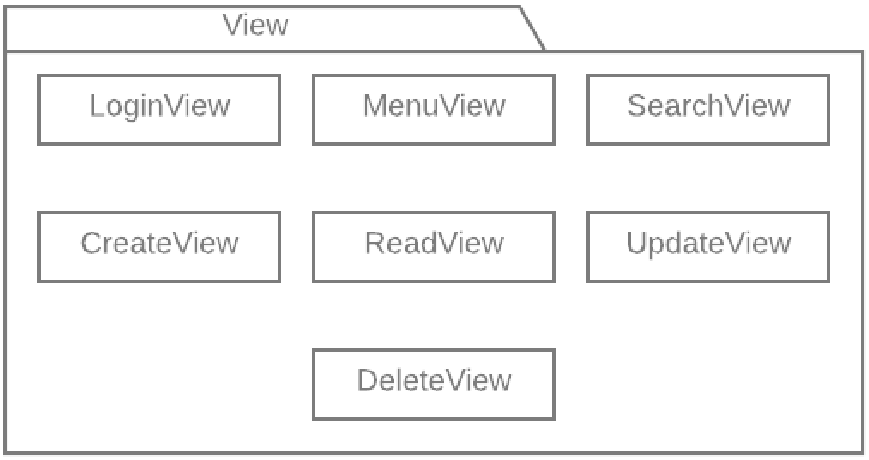
### 1.1.1. Model

The model component manages all the data-related logic, in other words, is the one that interacts with the database retrieving, deleting, updating, and creating new entries.
The model sends the data requested by the controller to the View.



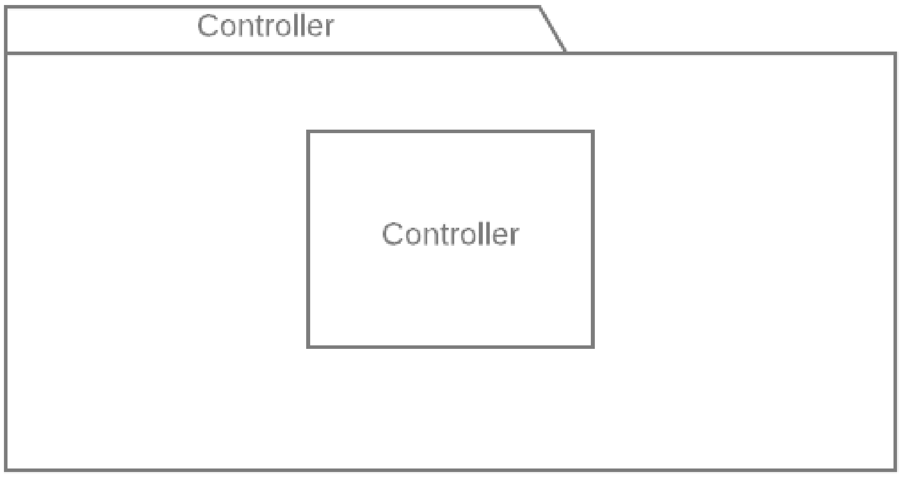
### 1.1.2. View

The View component is used for all the user interface logic of the application, the View presents to the user the information requested by the Controller to the Model.



### 1.1.3. Controller

The Controller component manages all the logic that integrates the View with the Model. The Controller manipulates the data using the Model and interacts with the View to render the new data.



### 1.2. Implement environment definition

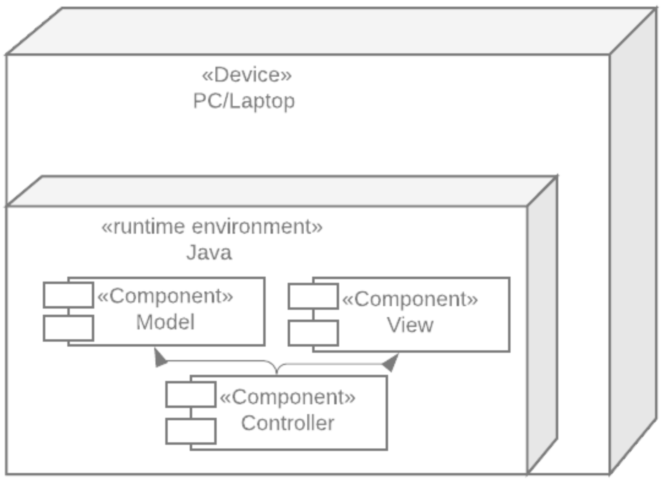
| Concept | Tool |
|---|---|
| Programming language | Java |
| Framework | Swing |
| UML diagram maker | Lucidchart and Intellij IDEA |
| Version control | Github |

## 2. Deployment view

### 2.1. Description

The one and only node on the system is the client.
Client: User computer with any of the most popular operating systems with java support (Windows, Linux, MacOs, etc).

### 2.2. Deployment model



# 3. Logic view

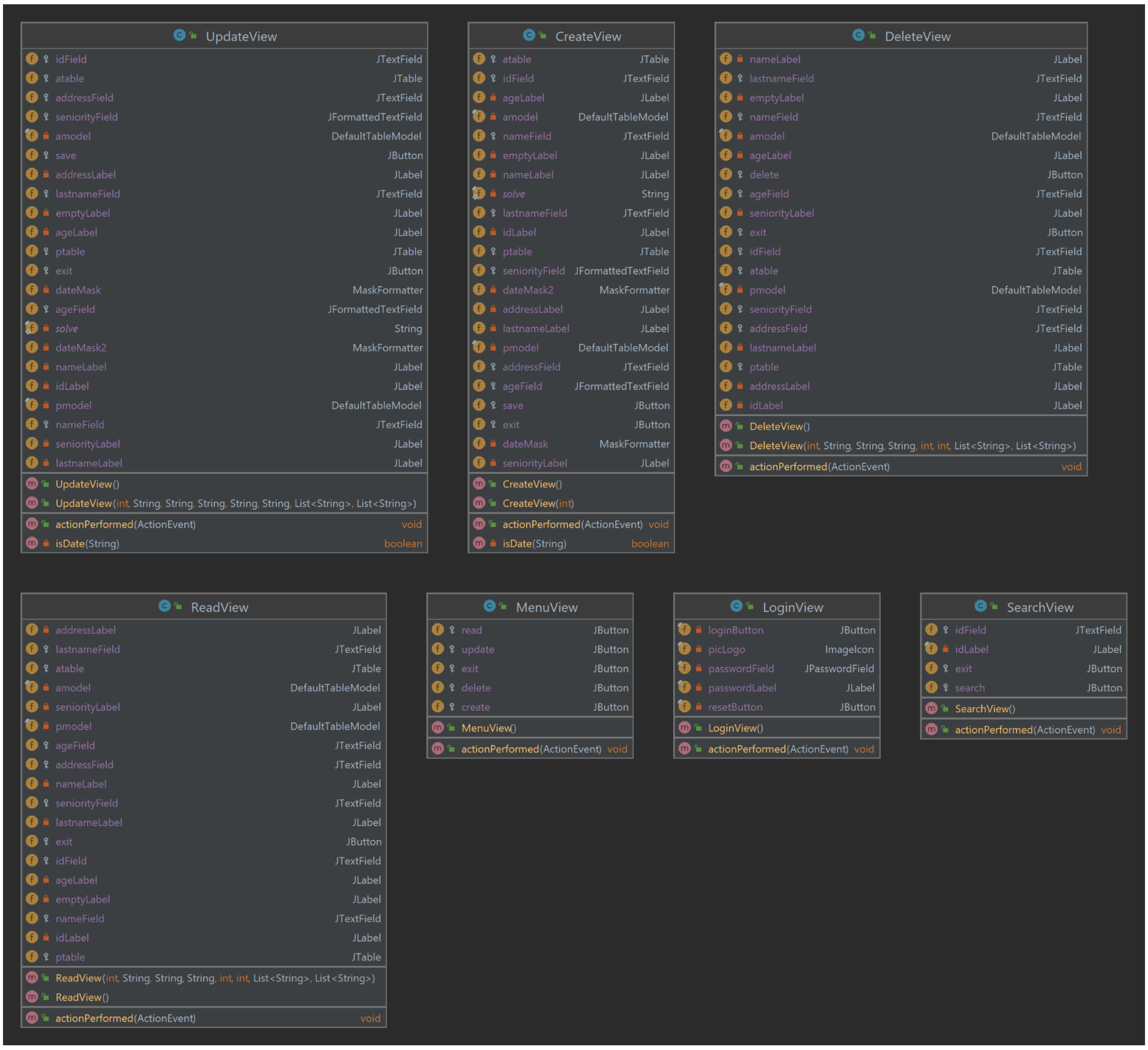
## 3.1. Description

This section contains the general composition of the classes of the application divided by packages.

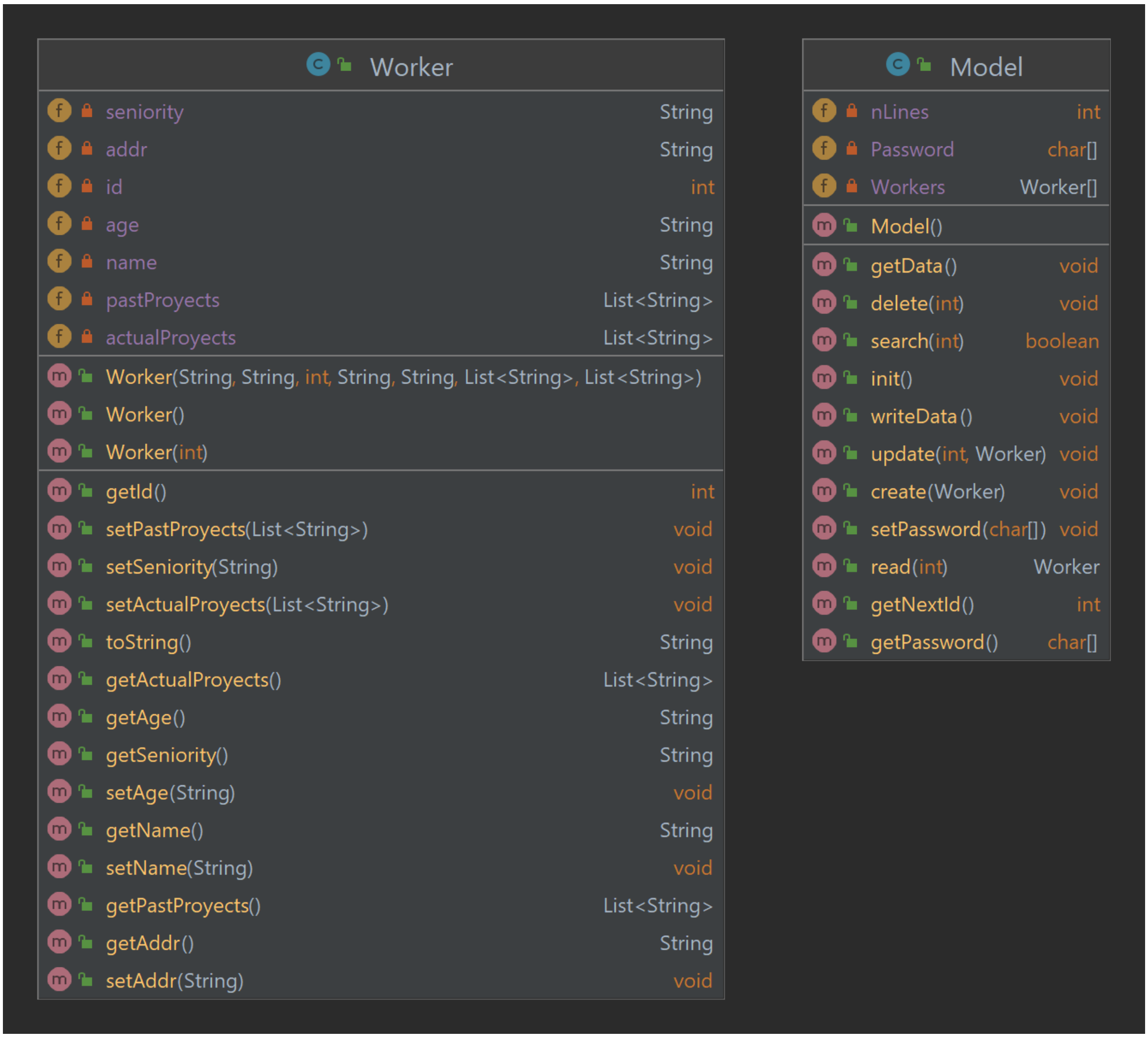
## 3.2. Class identification

3.3. This section contains the diagrams that describe the  behavior of the system and the relationships between the classes for all the packages.

### 3.3.1. View classes



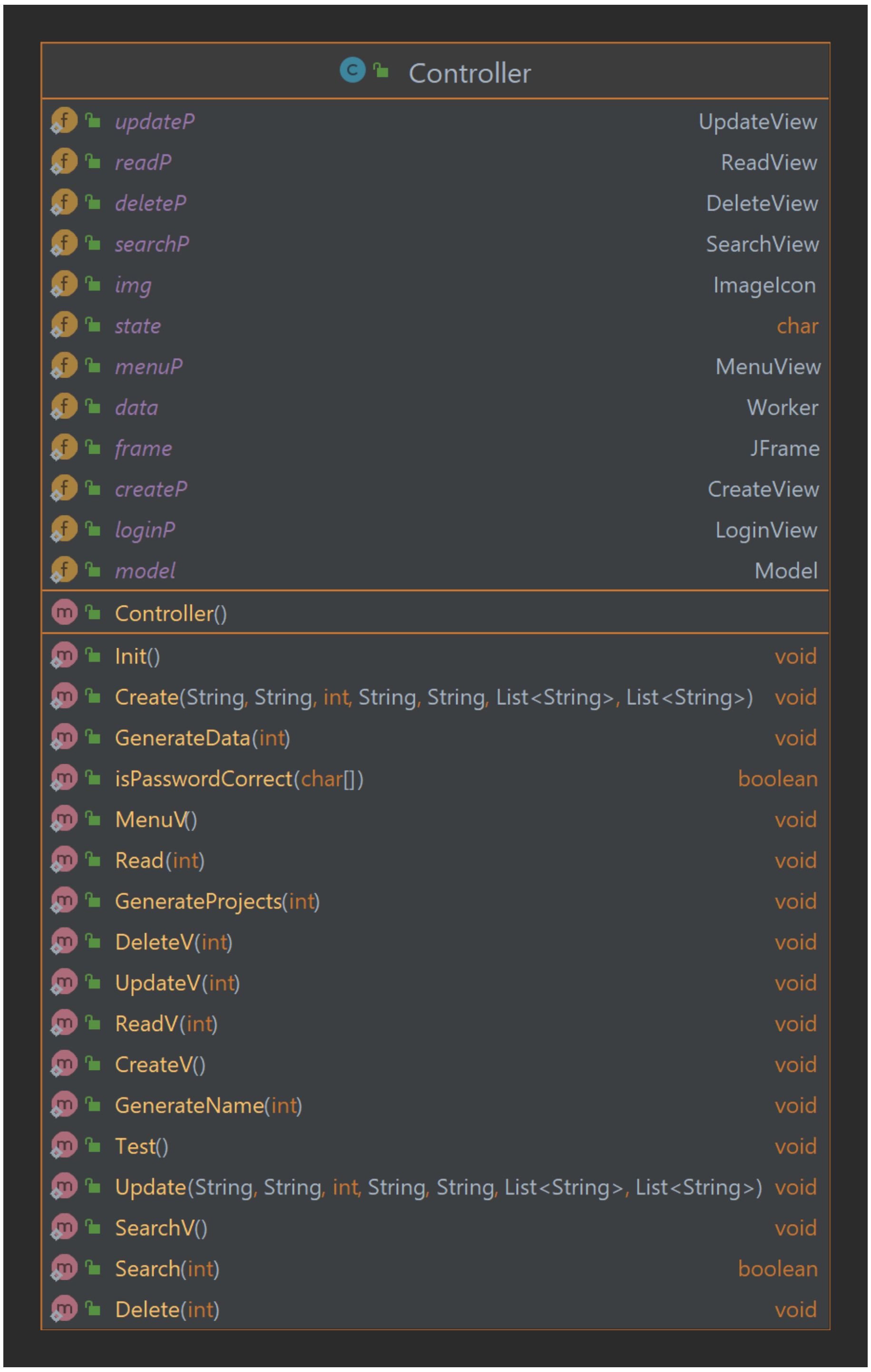
### 3.3.2. Model classes



| Worker | |
|---|---|
| seniority | String |
| addr | String |
| id | int |
| age | String |
| name | String |
| pastProyects | List<String> |
| actualProyects | List<String> |
| Worker(String, String, int, String, String, List<String>, List<String>) | |
| Worker() | |
| Worker(int) | |
| getId() | int |
| setPastProyects(List<String>) | void |
| setSeniority(String) | void |
| setActualProyects(List<String>) | void |
| toString() | String |
| getActualProyects() | List<String> |
| getAge() | String |
| getSeniority() | String |
| setAge(String) | void |
| getName() | String |
| setName(String) | void |
| getPastProyects() | List<String> |
| getAddr() | String |
| setAddr(String) | void |

| Model | |
|---|---|
| nLines | int |
| Password | char[] |
| Workers | Worker[] |
| Model() | |
| getData() | void |
| delete(int) | void |
| search(int) | boolean |
| init() | void |
| writeData() | void |
| update(int, Worker) | void |
| create(Worker) | void |
| setPassword(char[]) | void |
| read(int) | Worker |
| getNextId() | int |
| getPassword() | char[] |

### 3.3.3. Controller classes



Controller

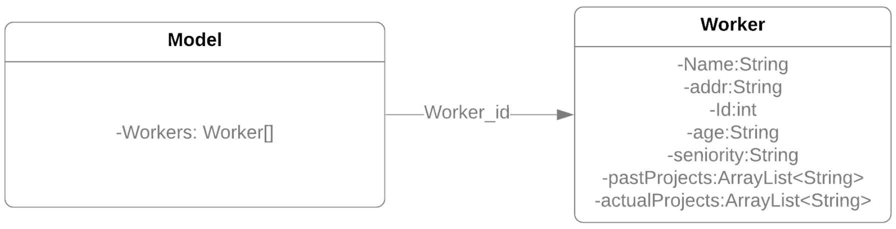
| | |
|---|---|
| updateP | UpdateView |
| readP | ReadView |
| deleteP | DeleteView |
| searchP | SearchView |
| img | ImageIcon |
| state | char |
| menuP | MenuView |
| data | Worker |
| frame | JFrame |
| createP | CreateView |
| loginP | LoginView |
| model | Model |
| Controller() | |
| Init() | void |
| Create(String, String, int, String, String, List<String>, List<String>) | void |
| GenerateData(int) | void |
| isPasswordCorrect(char[]) | boolean |
| MenuV() | void |
| Read(int) | void |
| GenerateProjects(int) | void |
| DeleteV(int) | void |
| UpdateV(int) | void |
| ReadV(int) | void |
| CreateV() | void |
| GenerateName(int) | void |
| Test() | void |
| Update(String, String, int, String, String, List<String>, List<String>) | void |
| SearchV() | void |
| Search(int) | boolean |
| Delete(int) | void |

# 4. Data view

## 4.1. Description

This section contains the diagram that describes the general data structure of the system.

## 4.2. Database diagram



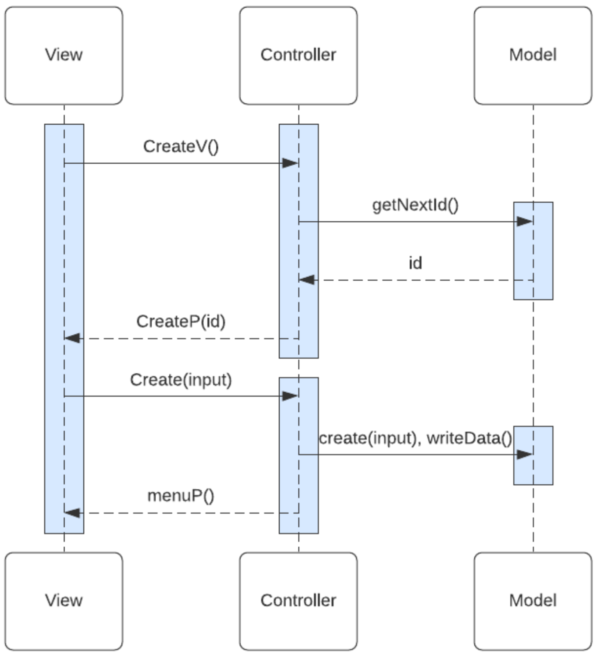
# 5. Dynamic view

## 5.1. Description

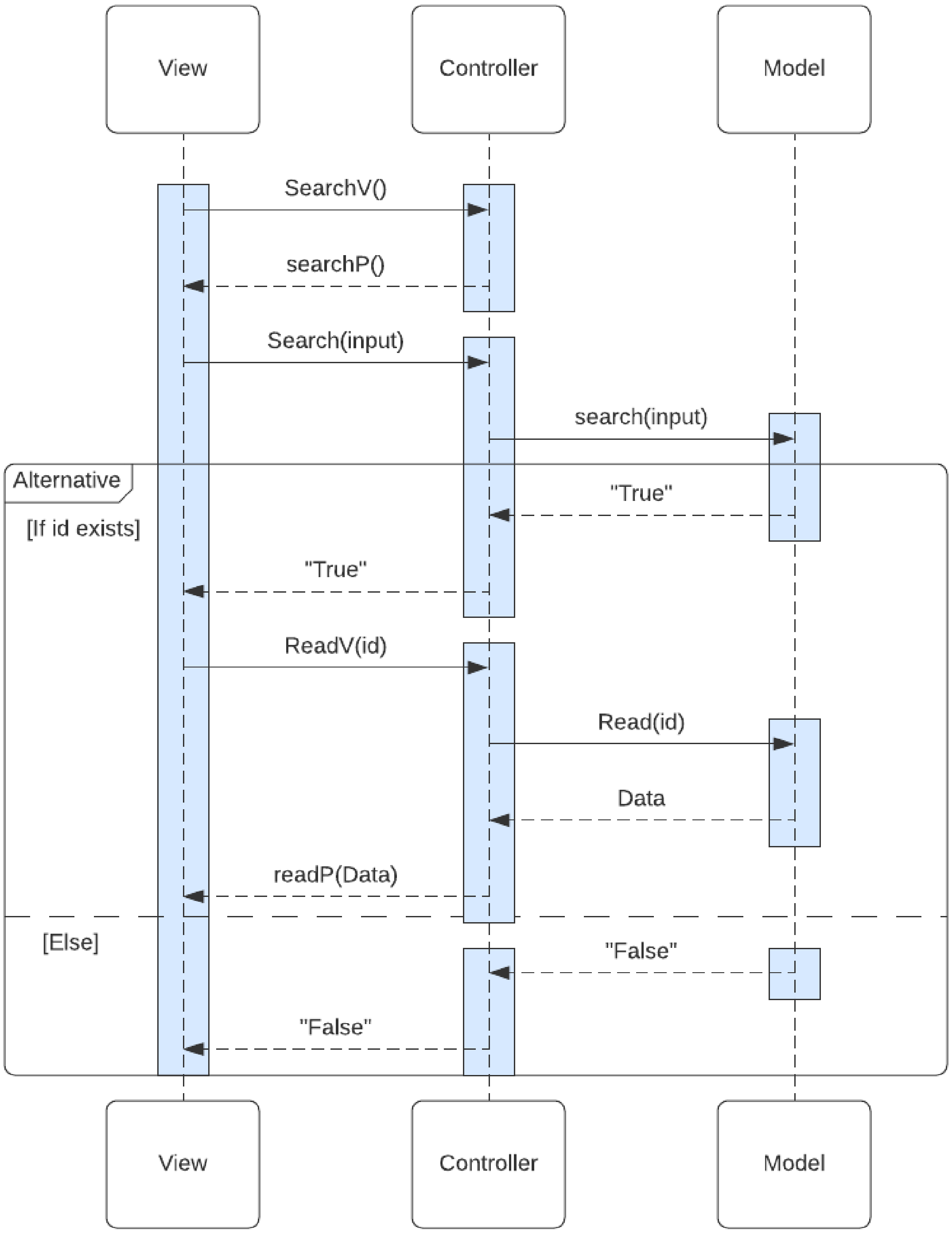
This section contains the sequence diagrams of all the use cases.
The diagrams show how the objects relate through the time.

## 5.2. Sequence diagrams

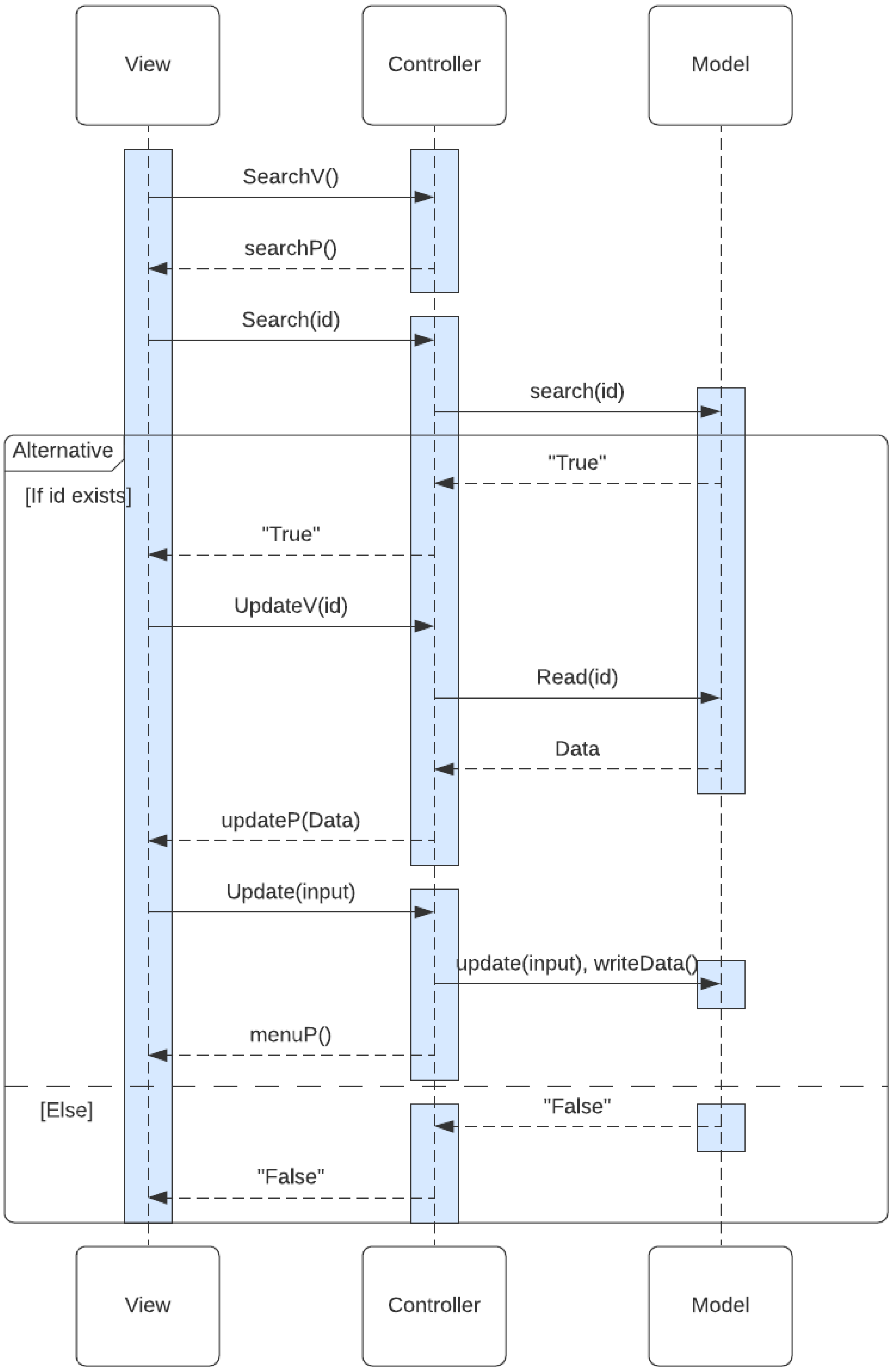
### 5.2.1. Create record



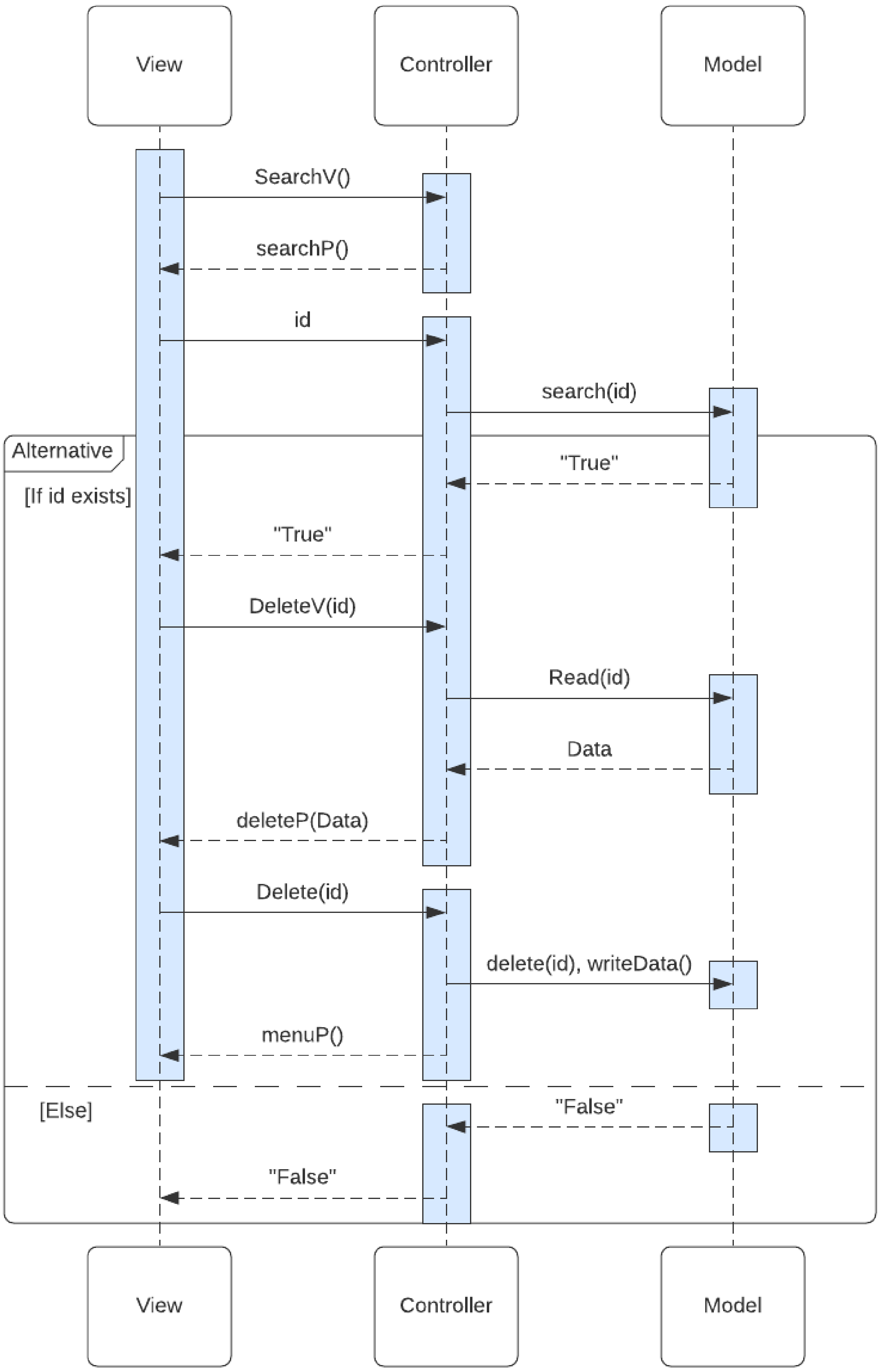
### 5.2.2. Read record



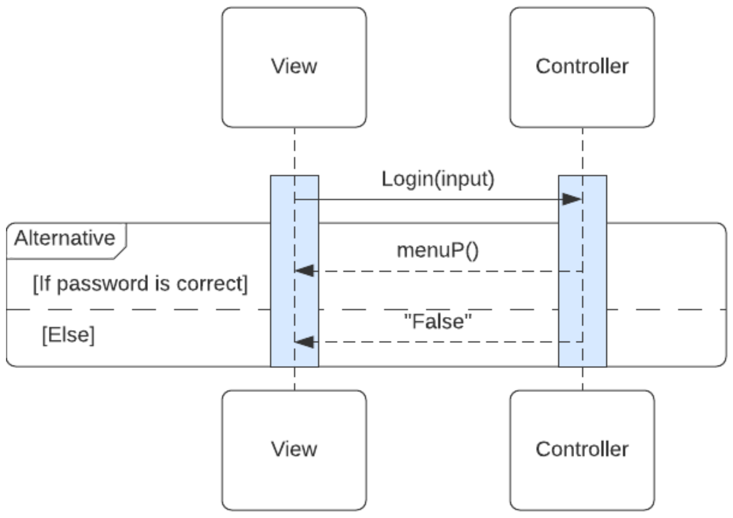
## 5.2.3. Update record



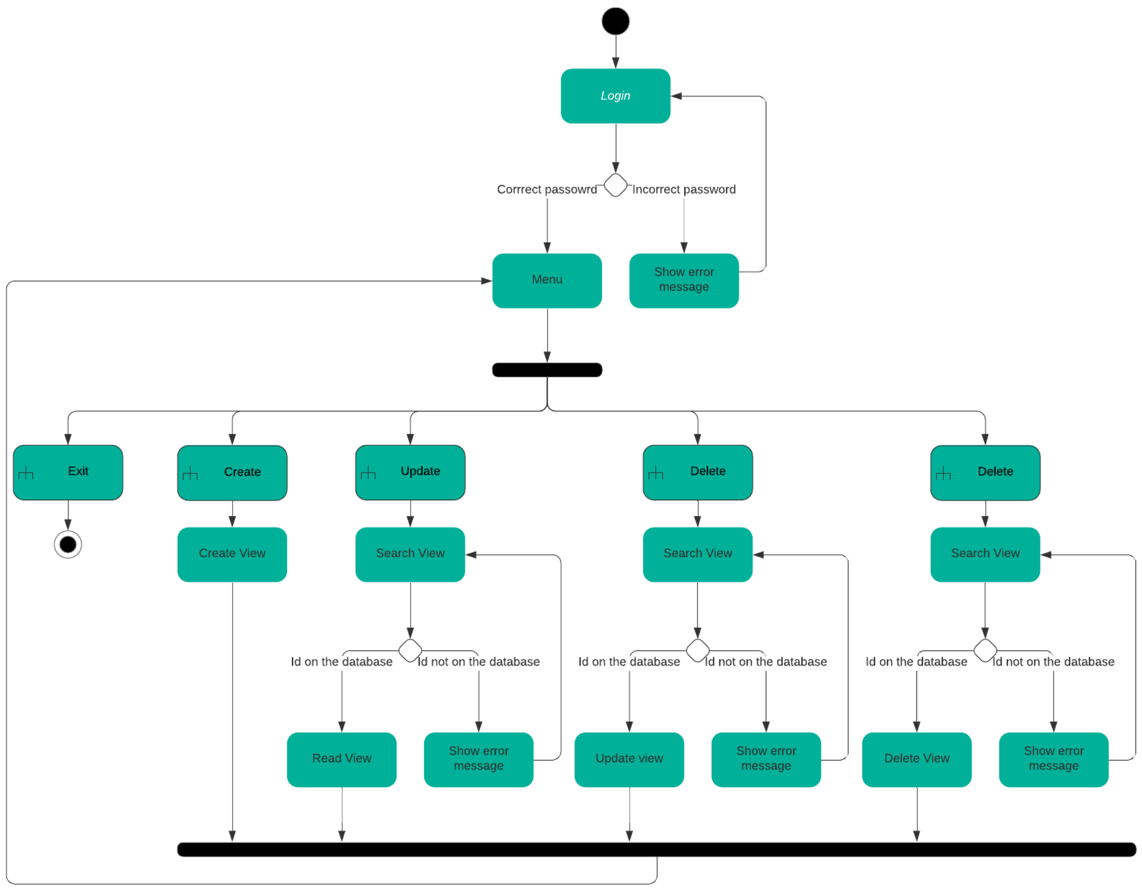
### 5.2.4. Delete record



### 5.2.5. Login



## 5.3. Navigation diagram


Login
Corrrect passowrd
Incorrect password
Menu
Show error message
Exit
Create
Update
Delete
Delete
Create View
Search View
Search View
Search View
Id on the database
Id not on the database
Id on the database
Id not on the database
Id on the database
Id not on the database
Read View
Show error message
Update view
Show error message
Delete View
Show error message