



**National Autonomous
University of Mexico**
School of Engineering
**Object-oriented programming
models**



Software Design

Version :1.0
9 of november 2021

Table of content:

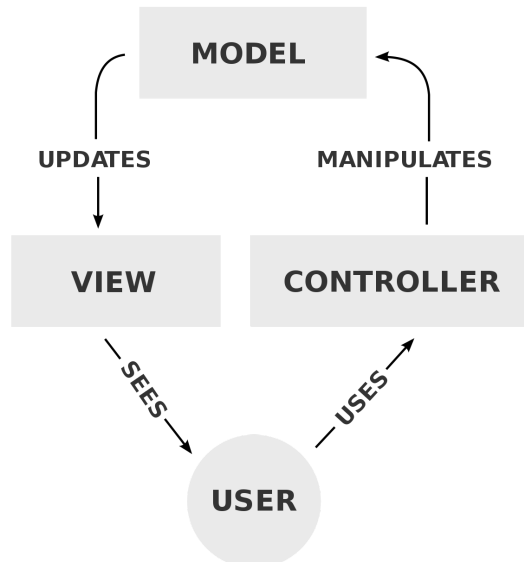
Changelog	3
Architecture description	4
Architecture packages	4
Model	4
View	5
Controller	5
Implement environment definition	6
Deployment view	6
Description	6
Deployment model	6
Logic view	7
Description	7
Class identification	7
View classes	7
Model classes	8
Controller classes	9
Data view	10
Description	10
Database diagram	10
Dynamic view	10
Description	10
Sequence diagrams	10
Create record	10
Read record	11
Update record	12
Delete record	13
Login	13
Navigation diagram	14

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

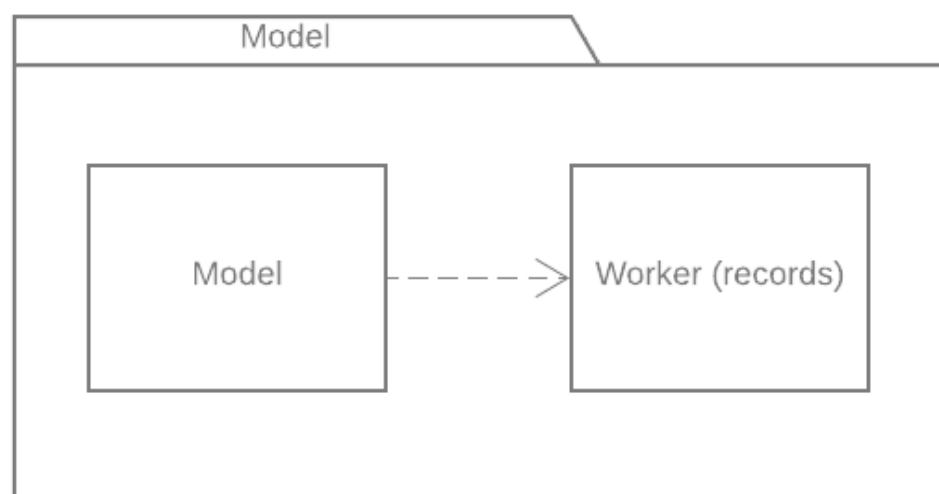


1.1. Architecture packages

1.1.1. Model

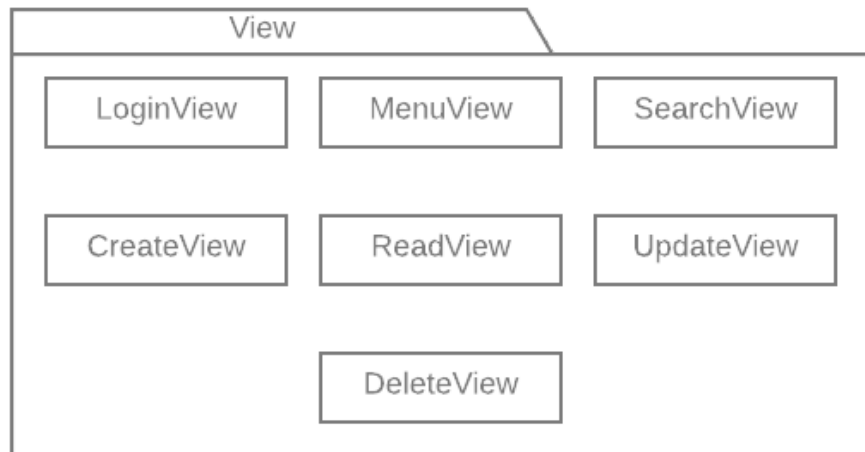
The model component manages all the data-related logic, in other words, is the one that interacts with the database retrieving, deleting, updating, and creating new entries.

The model sends the data requested by the controller to the View.



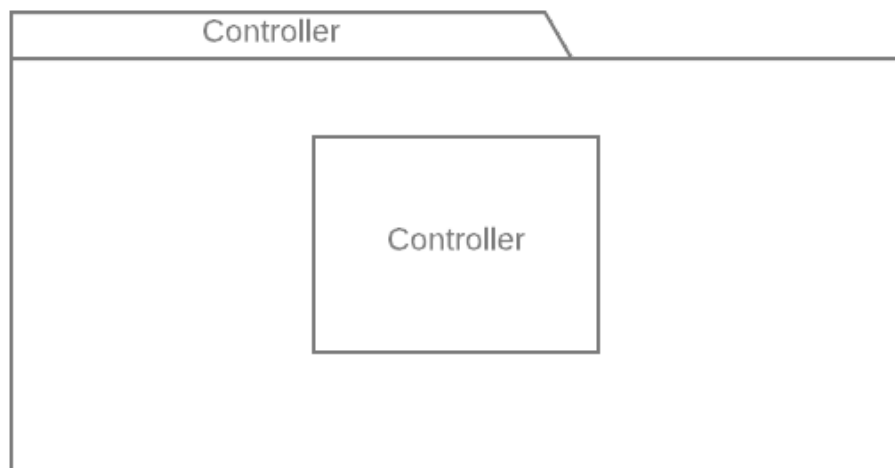
1.1.2. View

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

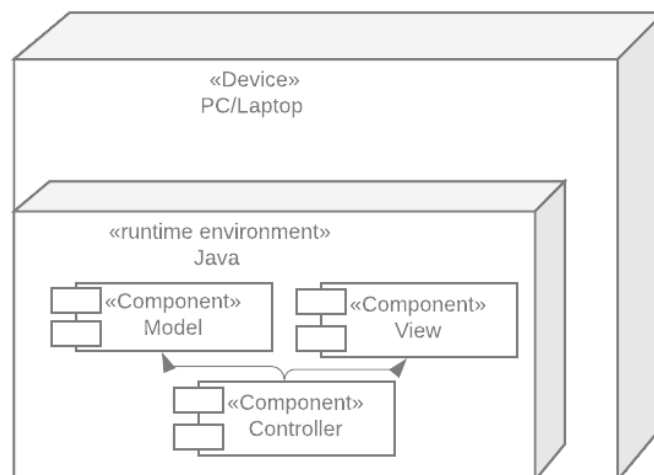
2. Deployment view

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

UpdateView	
idField	JTextField
atable	JTable
addressField	JTextField
seniorityField	JFormattedTextField
amodel	DefaultTableModel
save	JButton
addressLabel	JLabel
lastnameField	JTextField
emptyLabel	JLabel
ageLabel	JLabel
ptable	JTable
exit	JButton
dateMask	MaskFormatter
ageField	JFormattedTextField
solve	String
dateMask2	MaskFormatter
nameLabel	JLabel
idLabel	JLabel
pmodel	DefaultTableModel
nameField	JTextField
seniorityLabel	JLabel
lastnameLabel	JLabel
UpdateView()	
UpdateView(int, String, String, String, String, String, List<String>, List<String>)	
actionPerformed(ActionEvent)	void
isDate(String)	boolean

CreateView	
atable	JTable
idField	JTextField
ageLabel	JLabel
amodel	DefaultTableModel
nameField	JTextField
emptyLabel	JLabel
nameLabel	JLabel
solve	String
lastnameField	JTextField
idLabel	JLabel
ptable	JTable
seniorityField	JFormattedTextField
dateMask2	MaskFormatter
addressLabel	JLabel
lastnameLabel	JLabel
pmodel	DefaultTableModel
addressField	JTextField
ageField	JFormattedTextField
save	JButton
exit	JButton
dateMask	MaskFormatter
seniorityLabel	JLabel
CreateView()	
CreateView(int)	
actionPerformed(ActionEvent)	void
isDate(String)	boolean

DeleteView	
nameLabel	JLabel
lastnameField	JTextField
emptyLabel	JLabel
nameField	JTextField
amodel	DefaultTableModel
ageLabel	JLabel
delete	JButton
ageField	JTextField
seniorityLabel	JLabel
exit	JButton
idField	JTextField
atable	JTable
pmodel	DefaultTableModel
seniorityField	JTextField
addressField	JTextField
lastnameLabel	JLabel
ptable	JTable
addressLabel	JLabel
idLabel	JLabel
DeleteView()	
DeleteView(int, String, String, String, int, int, List<String>, List<String>)	
actionPerformed(ActionEvent)	void

ReadView	
addressLabel	JLabel
lastnameField	JTextField
atable	JTable
amodel	DefaultTableModel
seniorityLabel	JLabel
pmodel	DefaultTableModel
ageField	JTextField
addressField	JTextField
nameLabel	JLabel
seniorityField	JTextField
lastnameLabel	JLabel
exit	JButton
idField	JTextField
ageLabel	JLabel
emptyLabel	JLabel
nameField	JTextField
idLabel	JLabel
ptable	JTable
ReadView(int, String, String, String, int, int, List<String>, List<String>)	
ReadView()	
actionPerformed(ActionEvent)	void

MenuView	
read	JButton
update	JButton
exit	JButton
delete	JButton
create	JButton
MenuView()	
actionPerformed(ActionEvent)	void

LoginView	
loginButton	JButton
picLogo	ImageIcon
passwordField	JPasswordField
passwordLabel	JLabel
resetButton	JButton
LoginView()	
actionPerformed(ActionEvent)	void



SearchView	
idField	JTextField
idLabel	JLabel
exit	JButton
search	JButton
SearchView()	
actionPerformed(ActionEvent)	void

3.3.2. Model classes

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

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

3.3.3. Controller classes

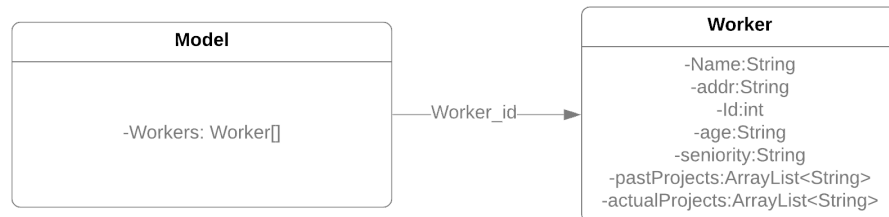
Controller		
  <i>updateP</i>		UpdateView
  <i>readP</i>		ReadView
  <i>deleteP</i>		DeleteView
  <i>searchP</i>		SearchView
  <i>img</i>		Imagelcon
  <i>state</i>		char
  <i>menuP</i>		MenuView
  <i>data</i>		Worker
  <i>frame</i>		JFrame
  <i>createP</i>		CreateView
  <i>loginP</i>		LoginView
  <i>model</i>		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



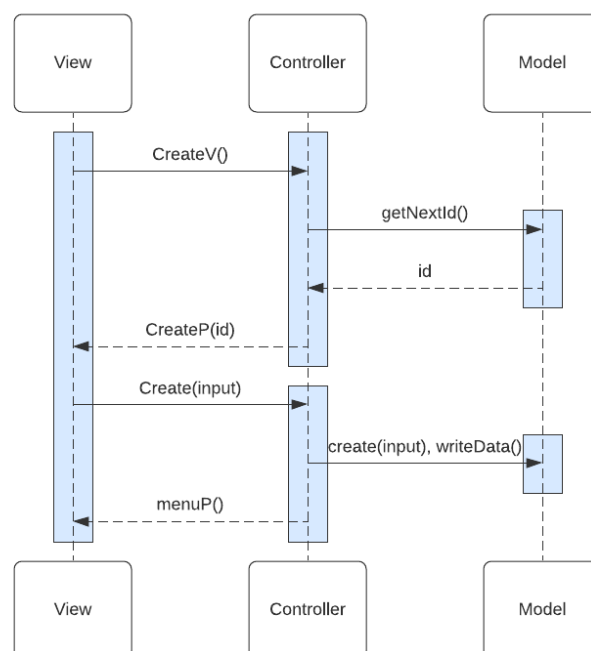
5. Dynamic view

5.1. Description

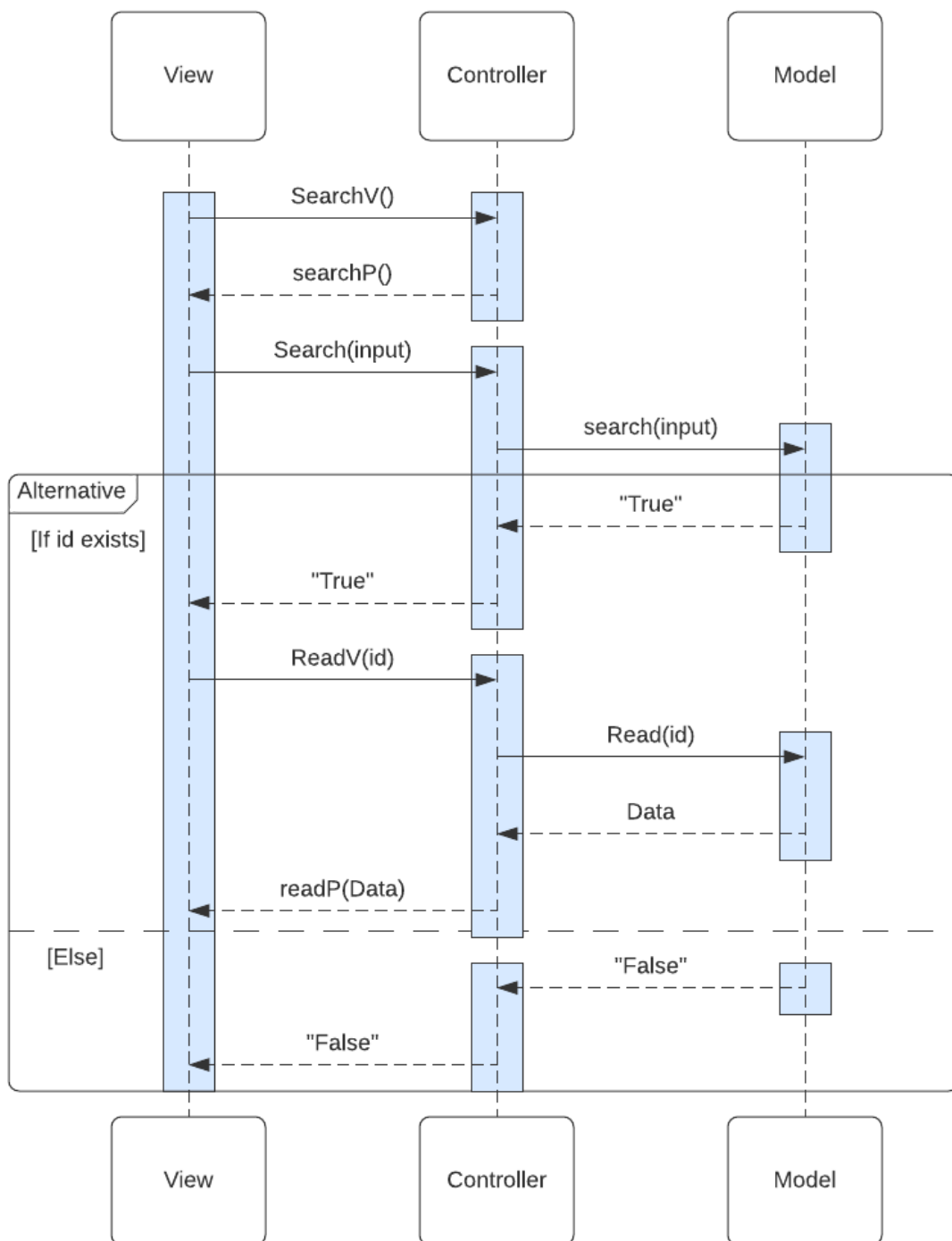
This section contains the sequence diagrams of all the use cases. The diagrams show how the objects relate through the time.

5.2. Sequence diagrams

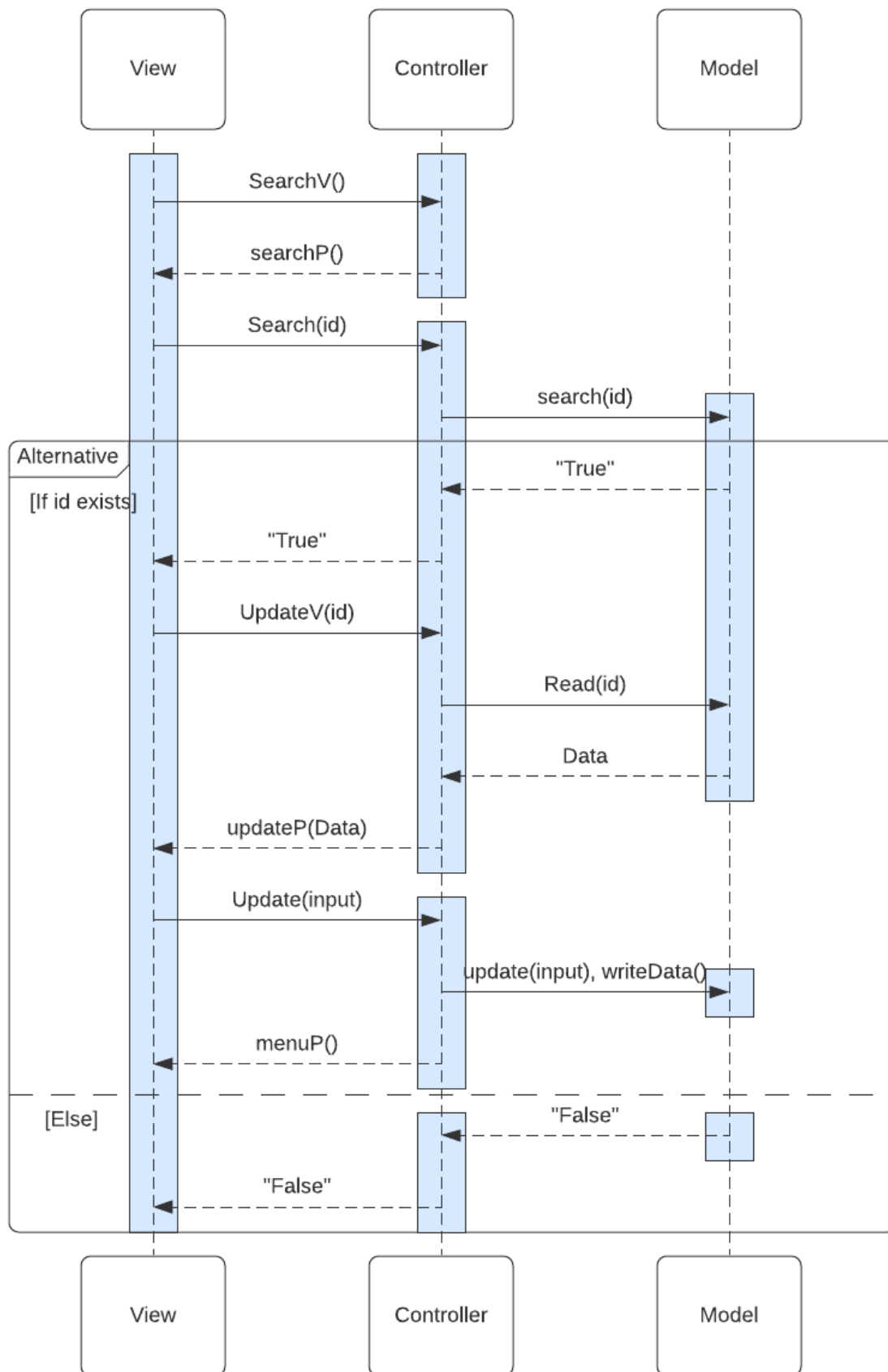
5.2.1. Create record



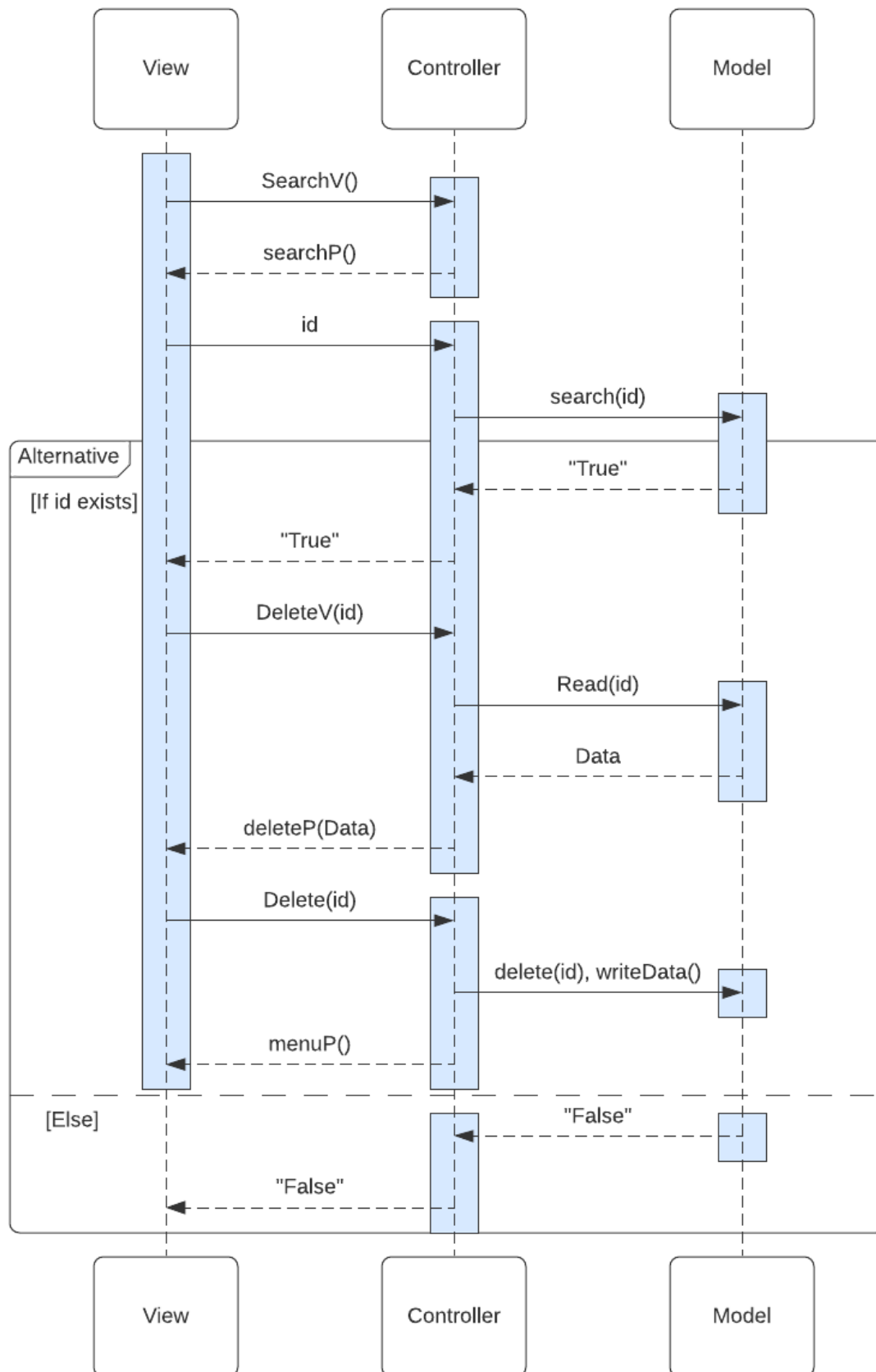
5.2.2. Read record



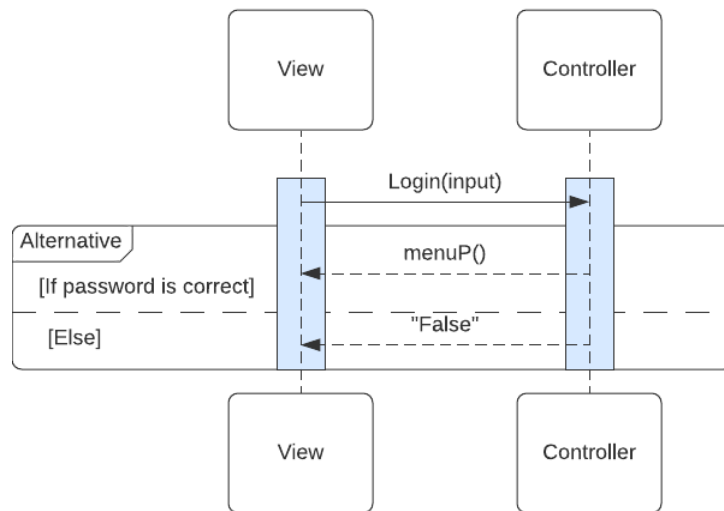
5.2.3. Update record



5.2.4. Delete record



5.2.5. Login



5.3. Navigation diagram

