



ADIV
Job Searching

By

Nuttapat Pimthong	59010444
Nuttasit Boonsai	59010484
Bundit Seedao	59010759
Settachat Tungpitagkai	59011345

Present

Asst.Prof.Dr. Visit Hirankitti

Subject Object-Oriented Analysis and Design
Department of Computer Engineering. Faculty of Engineering.
King Mongkut's Institute of Technology Ladkrabang
Chapter I

Introduction and Requirement Specification

Functional requirement

Account System

Register System

1. User: User must fill in email & password for register an account.

- E-mail
- Password

2. choose one chiod between seeker or company profile.

Login System

Fill-in e-mail and password for authentication.

Profile System

Can create & update account.

1. Seeker Type

- Name - Surname
- Telephone Number
- Address
- Education Level
- Biography

2. Company Type

- Company Name
- Telephone Number
- Industry
- Address
- Website
- Biography

Post Job System

Company can post requirement of work.

- Employment type
- Career level
- Job Function
- Salary
- Location
- Skill

Searching System

Search for work type.

- Job Function
- Employment type
- Career level
- Salary Range

- Location
- Skill

Matching Job System

Apply Job

When job seeker apply for a job .it can be displayed to the company.

Approve Job

company can receive or reject job seeker. For interviewers to get the results.

Database System

Save all object to one file and store all object when start program.

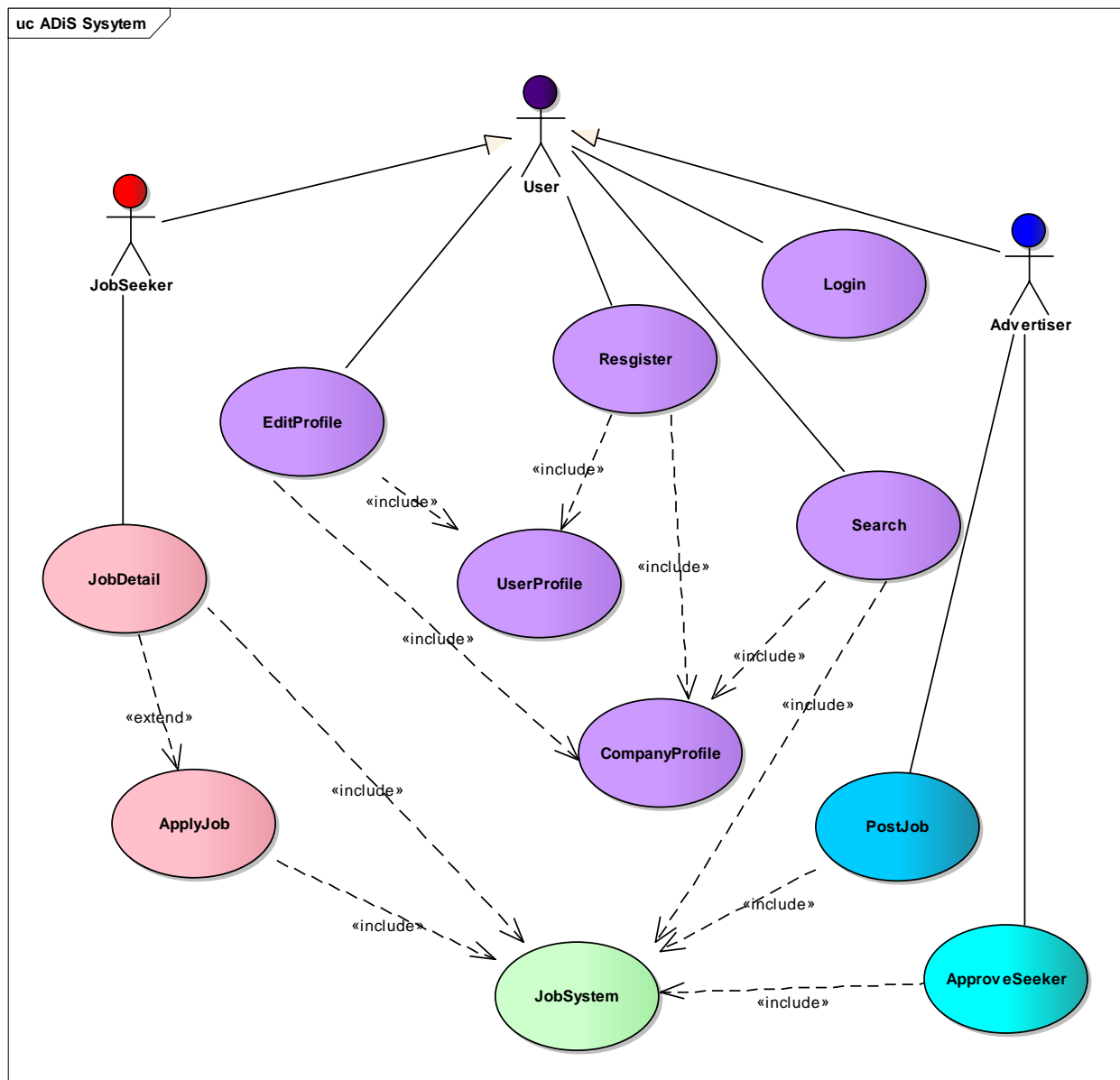
Non-Functional requirement

- Performance: มีการทดสอบประสิทธิภาพด้านเวลาในการตอบสนอง
- Reliability: มีการทดสอบระบบก่อนใช้งานจริง
- Usability: ความง่ายในการใช้งาน ด้วย interface ที่ดี
- Testability: ความสามารถในการทดสอบ
- Modifiability: ความสามารถในการปรับปรุงแก้ไขได้มีความยืดหยุ่น
ผลกระทบข้างเคียงน้อย

Chapter II

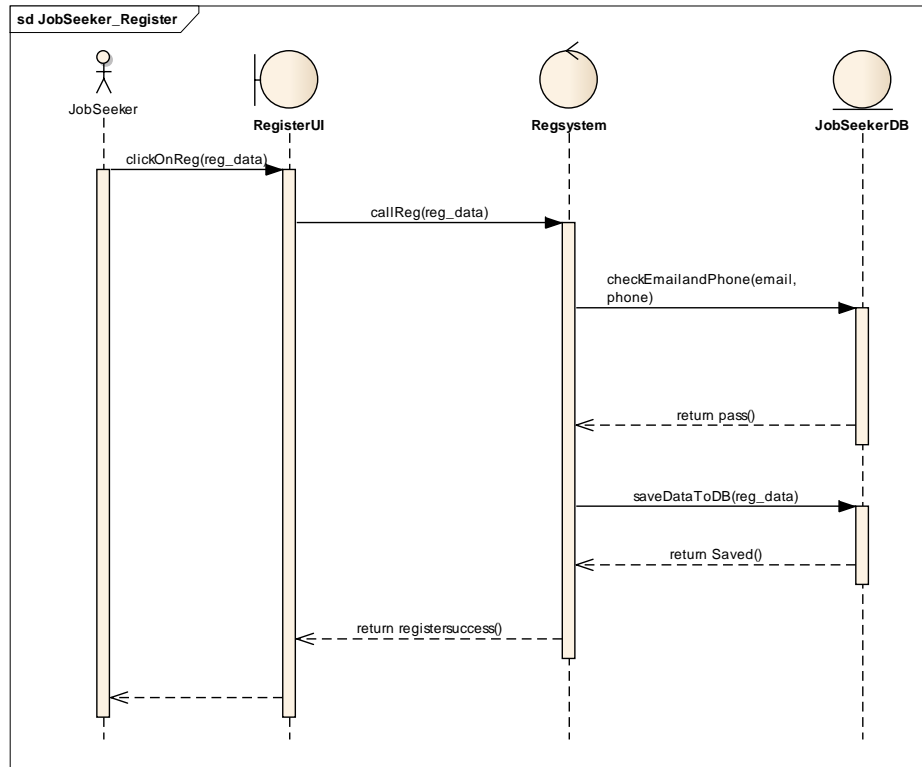
Analysis

Use Case diagram



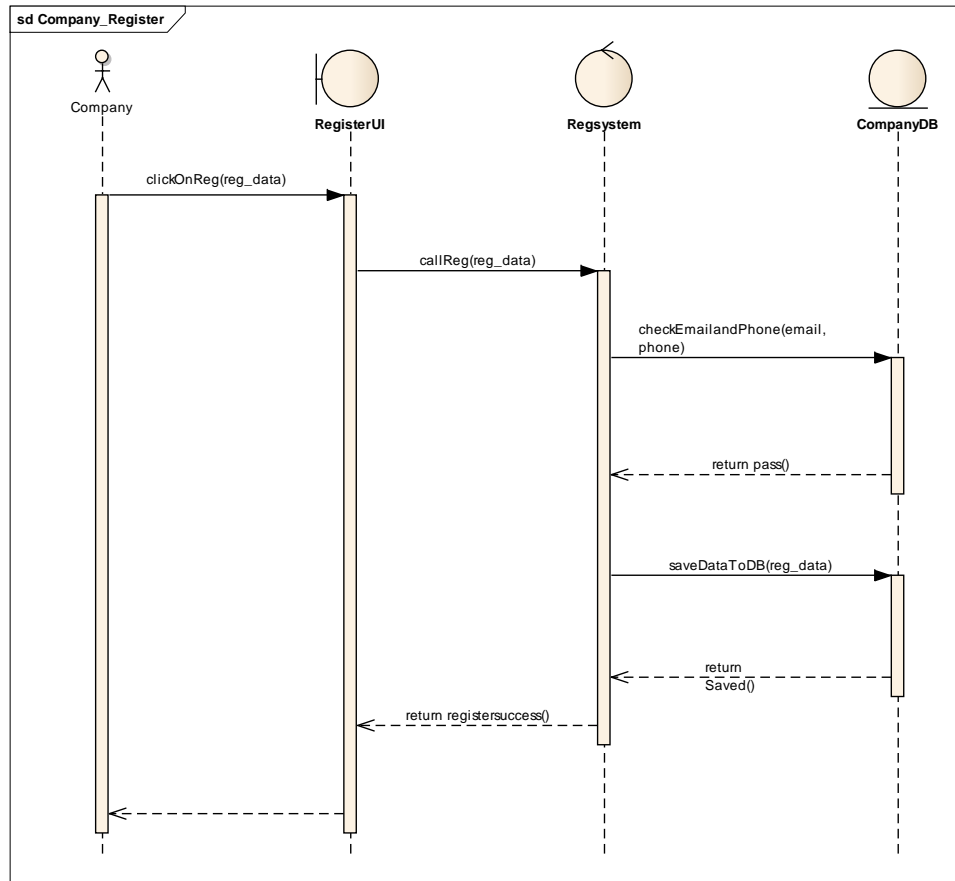
Sequence diagram

Register



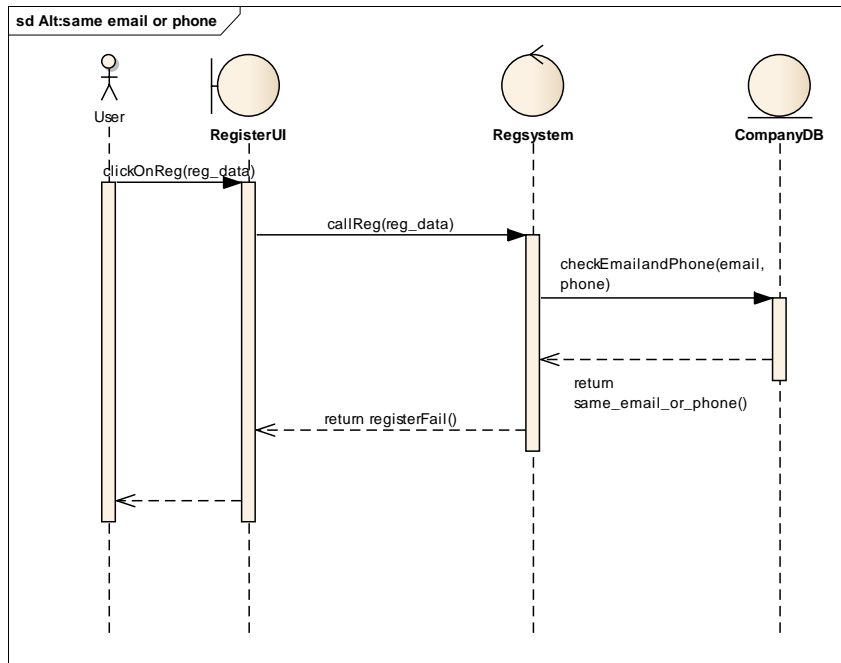
Basic course of event 1:

When job seeker fill in details to creates an account and clicks on register. Register UI will transfer data to Register System for manage about register and check email and phone for alternative case (not same email and phone with JobSeekerDB) then save to JobSeekerDB and update UI.



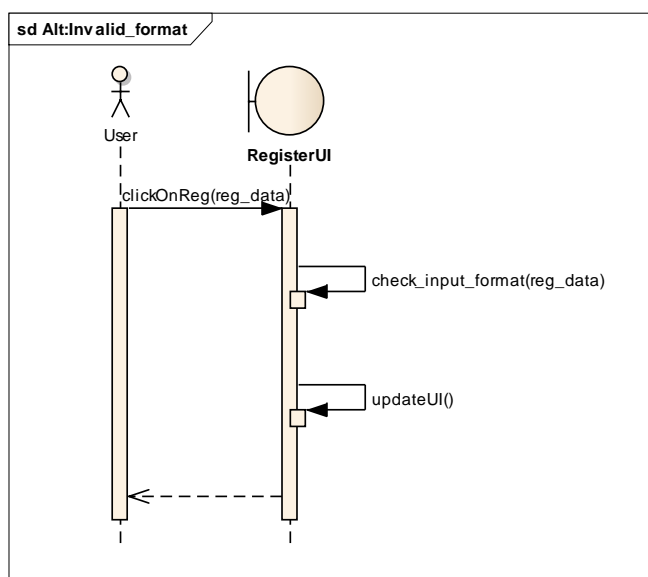
Basic course of event 2:

In this part for company has same flow with Job seeker but kept data in companyDB



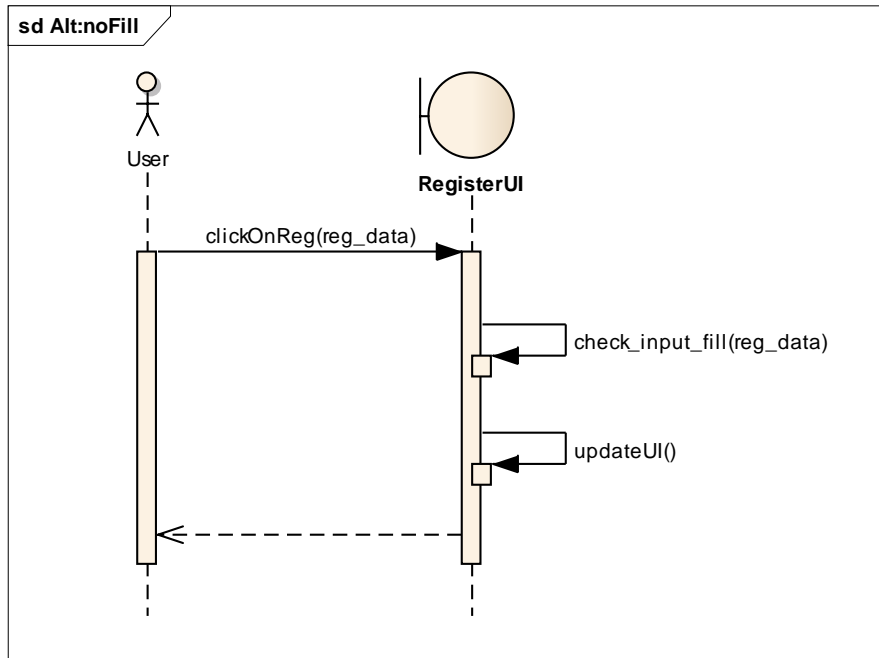
Alternative course of event 1:

If the user enter the email or phone number used to be used by other users in the system, the Register System will review and send back the UI to update the application is not successful.



Alternative course of event 2:

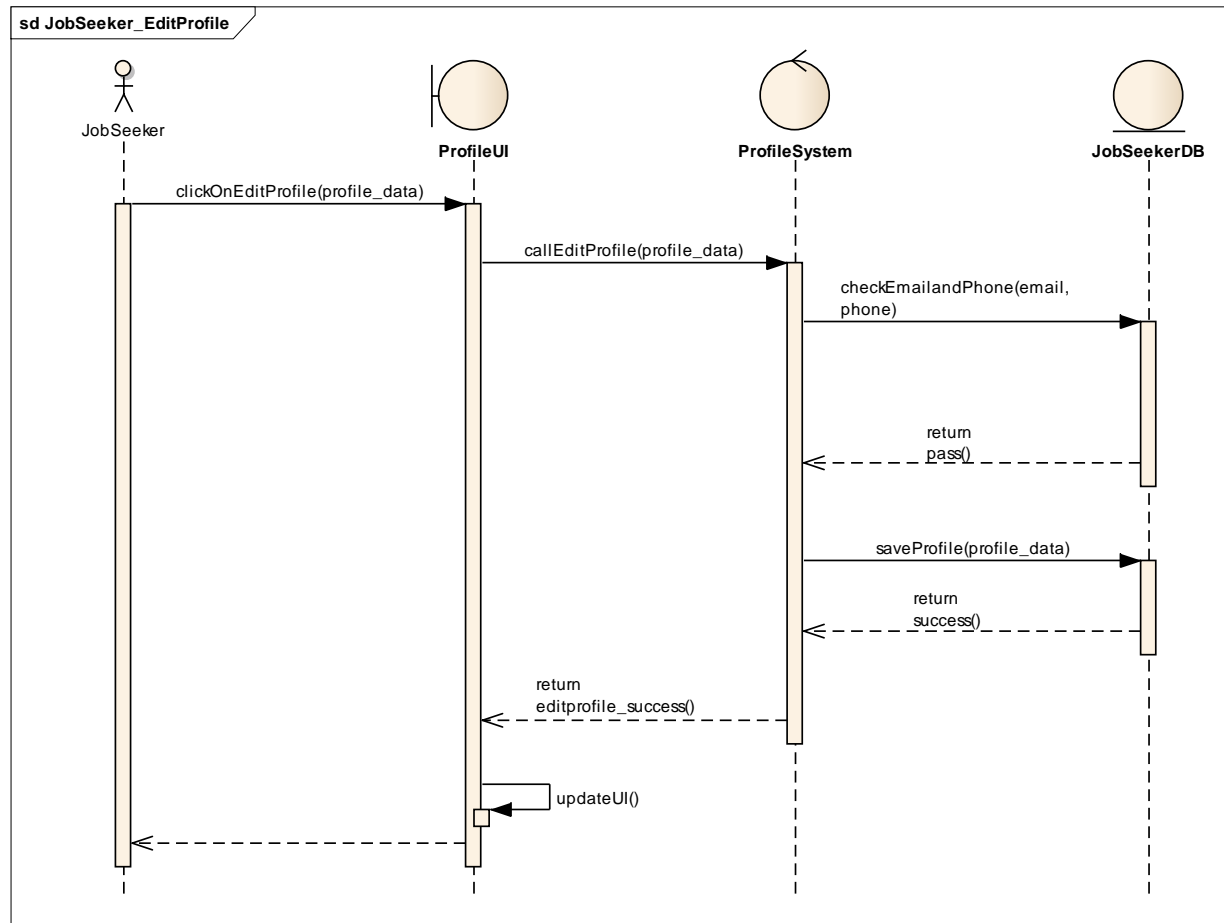
If a user types in the wrong format, such as the wrong form of email. The UI will perform a check before sending the value to the Register System, updating the UI to indicate that the data format is wrong.



Alternative course of event 3:

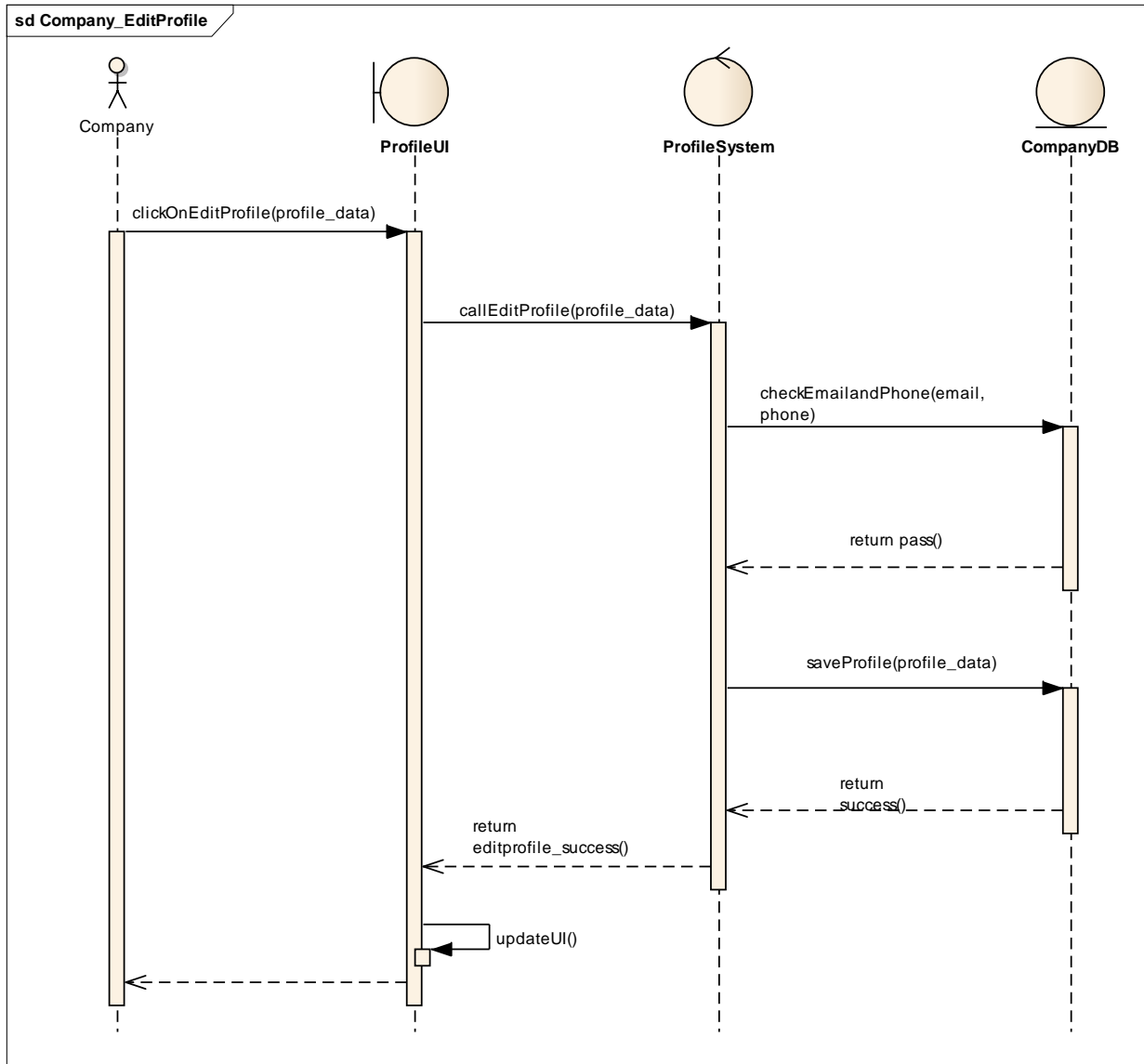
If the user does not fill in some fields, the UI will check and update the UI to indicate that some fields are missing.

UpdateProfile



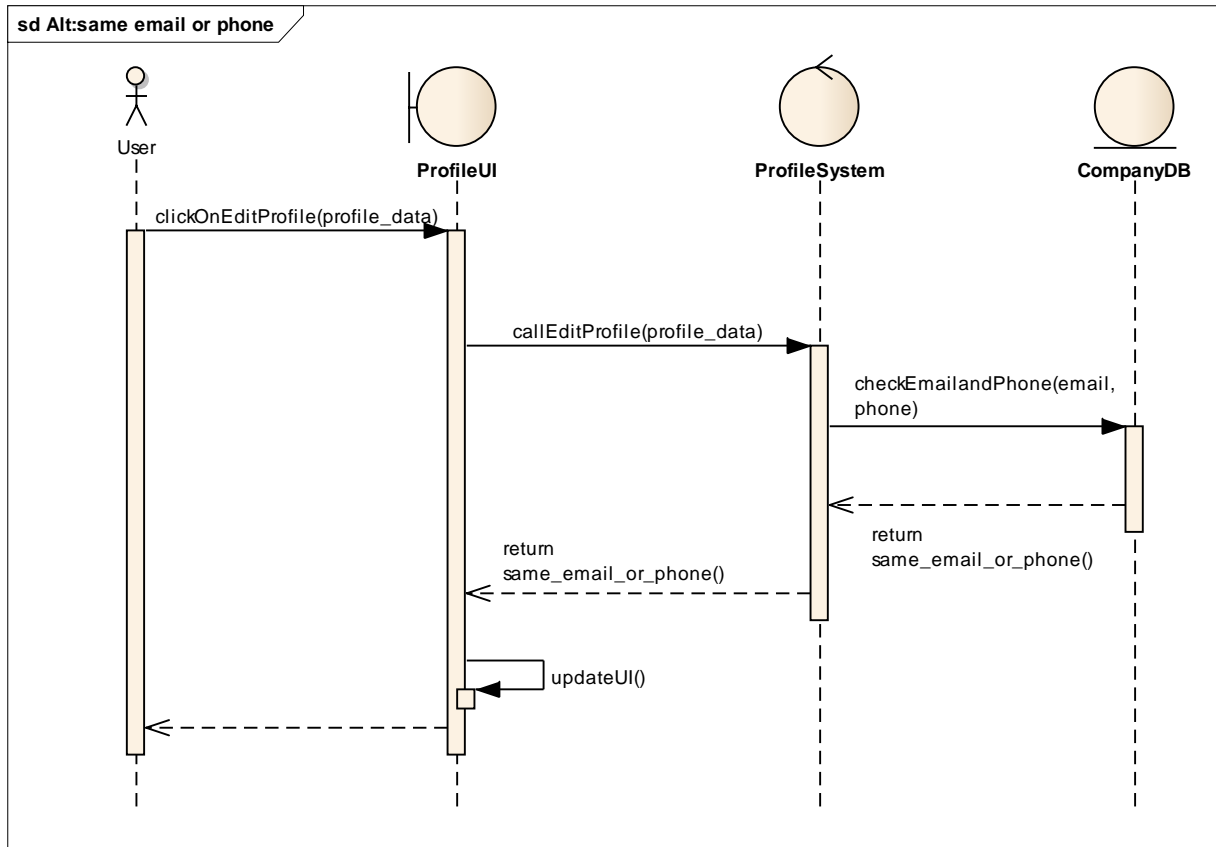
Basic course of event 1:

When a job seeker corrects a profile And then press Update. The system will send information to the system edit profile as a manager to edit the profile information. This will check email, phone with user data type. When validated, it will record data. And send back the information. UI to update the interface.



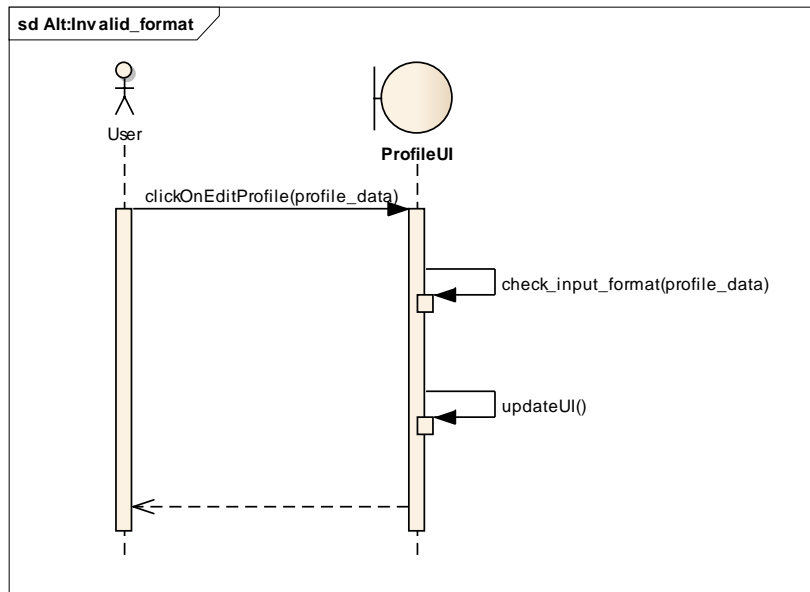
Basic course of event 2:

This is the part of the company whose flow of work will be like. Of the nearly all job seekers. Except for kept data in companyDB.



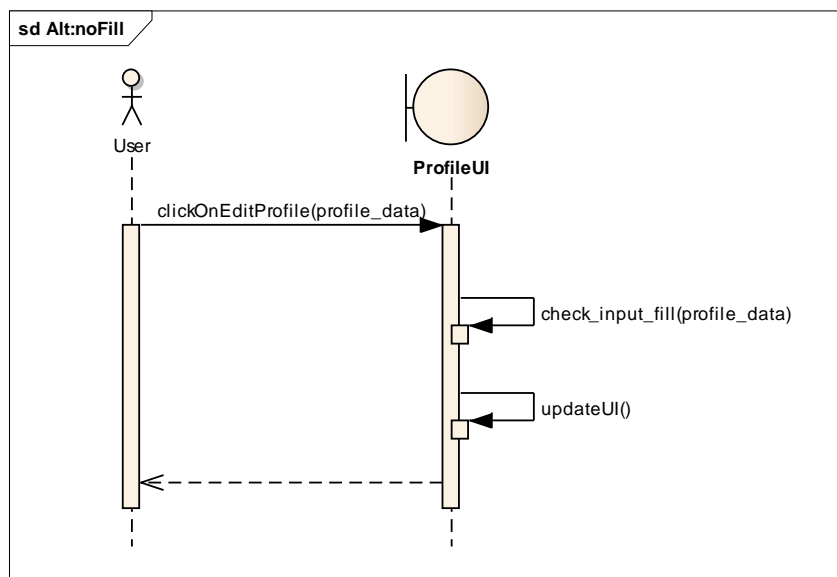
Alternative course of event 1:

If the user enter the email or phone number used by another user in the system, the profile system will be checked and sent back to ProfileUI.



Alternative course of event 2:

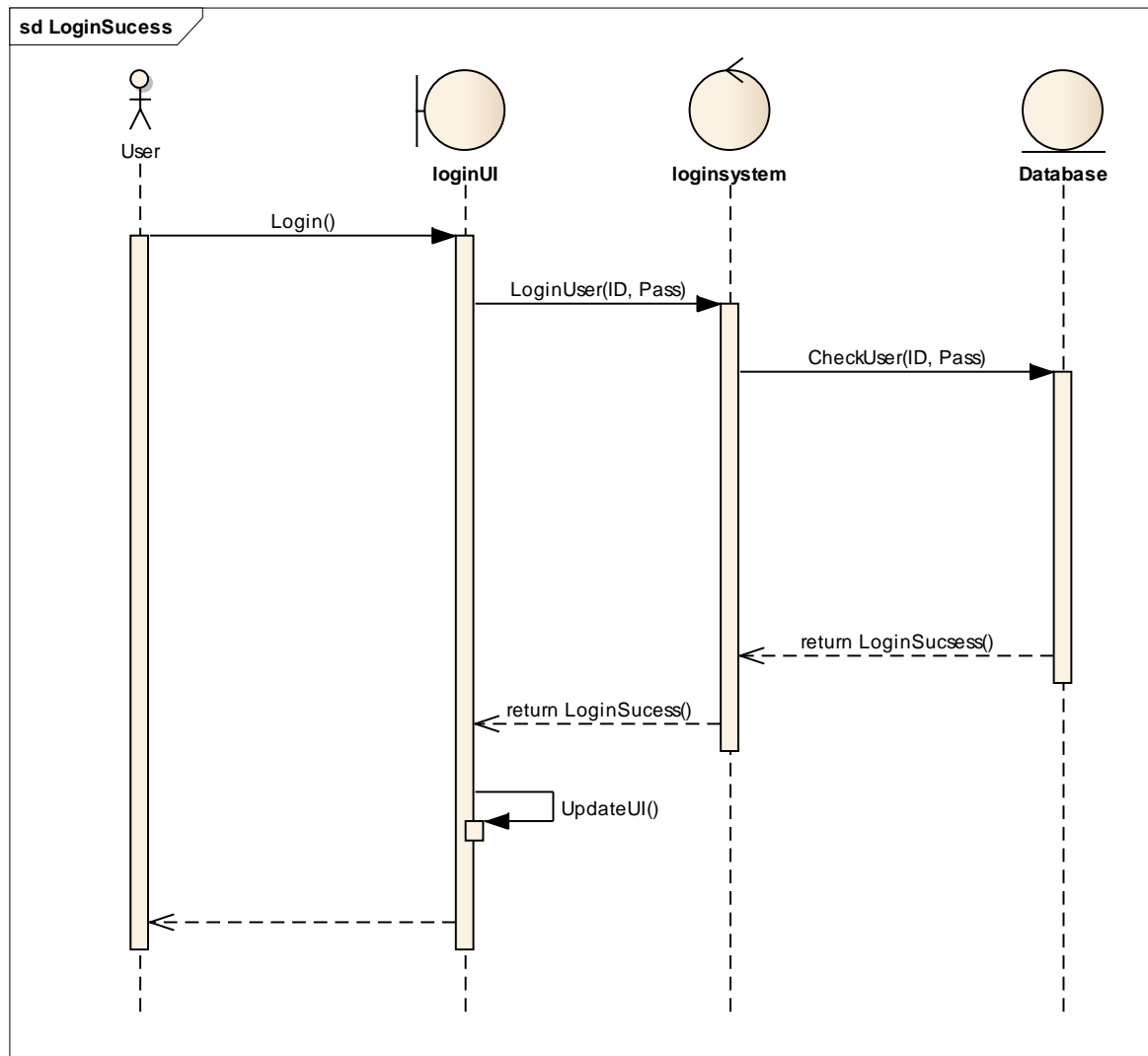
If a user types in the wrong format, such as the wrong form. The ProfileUI will check before sending the configuration to the sys profile system. It will update UI to indicate that the data format is wrong.



Alternative course of event 3:

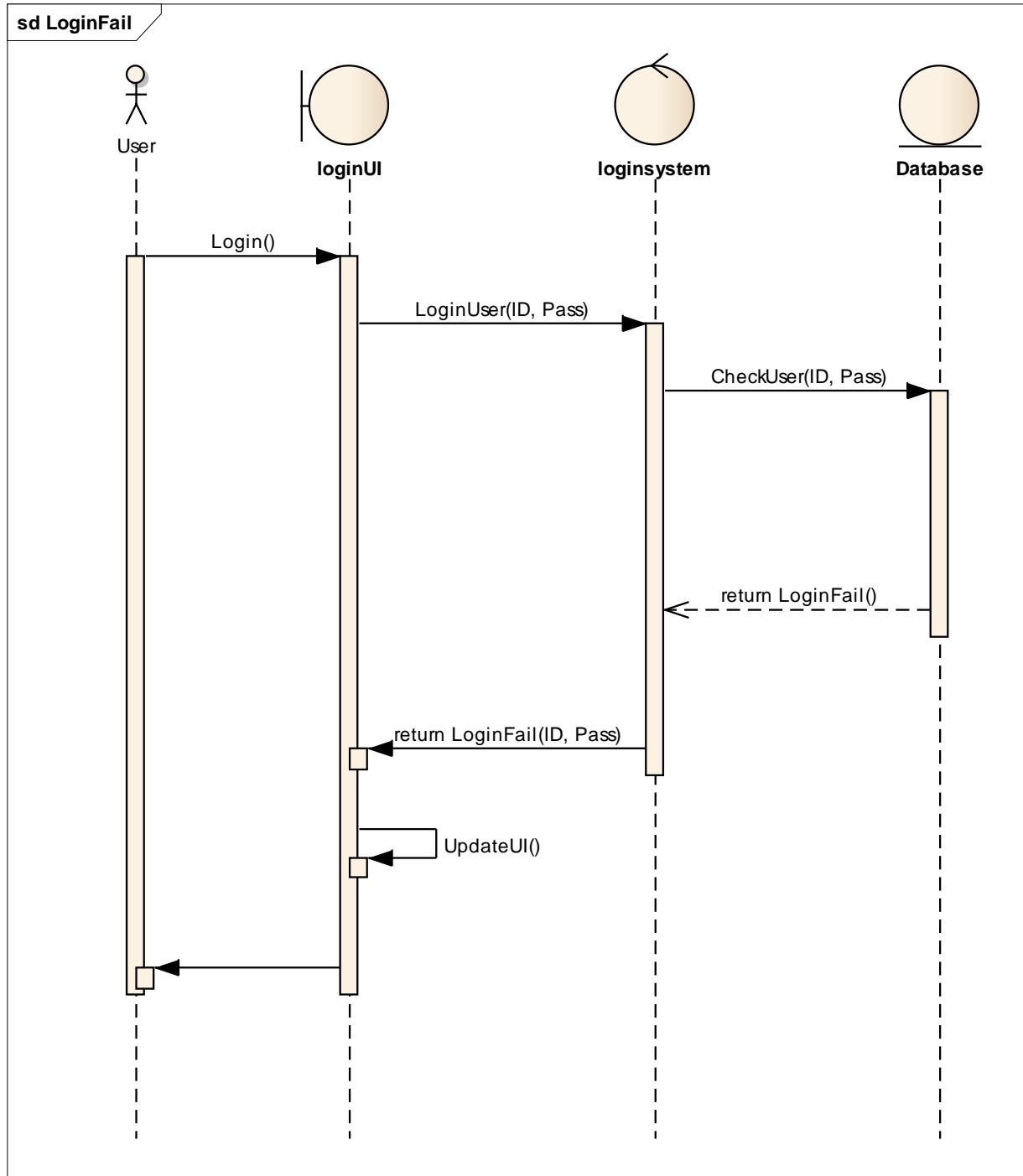
If the user does not fill in the required fields, the system ProfileUI will check and update UI to indicate that some fields are missing.

Login



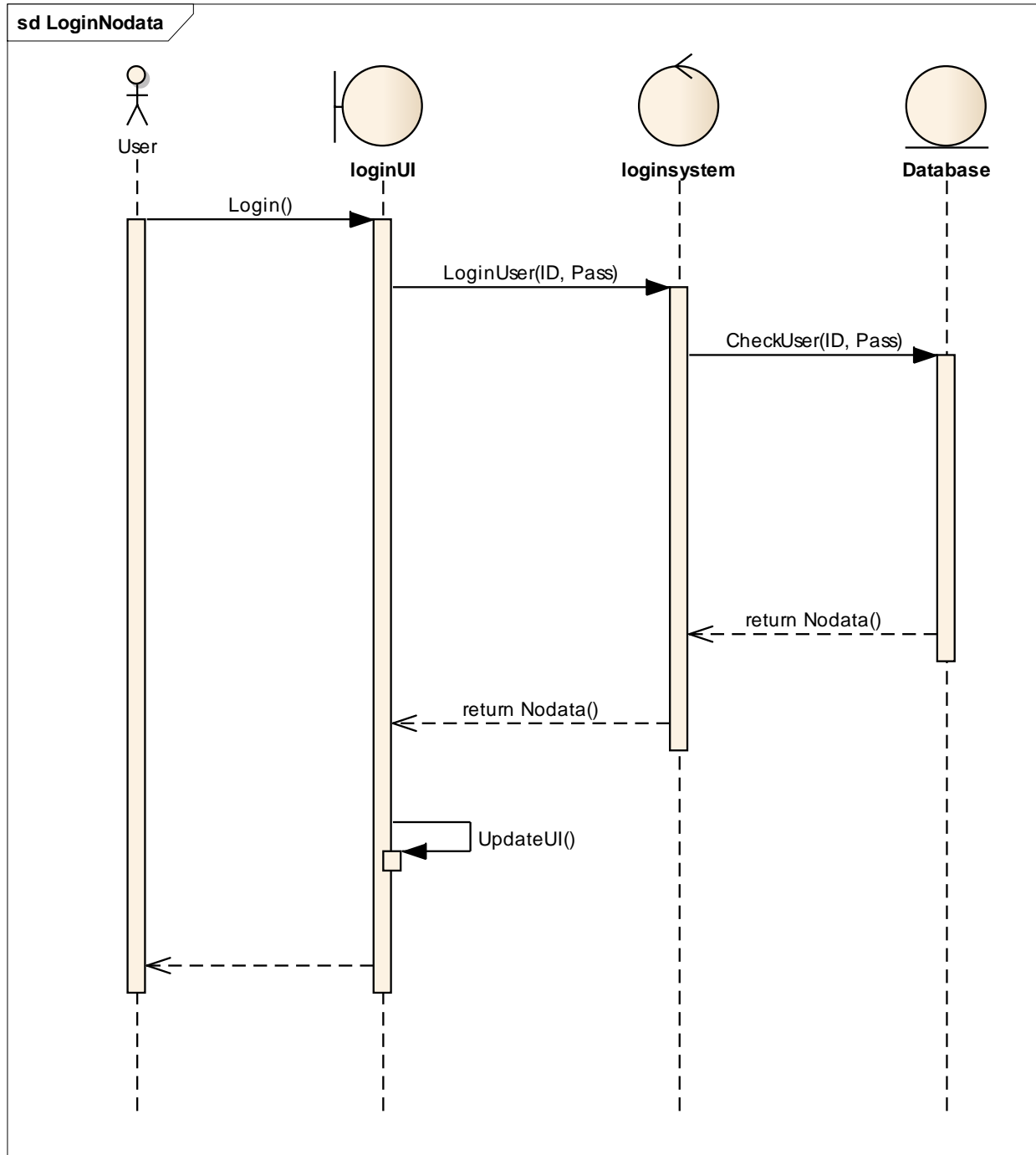
Basic course of event:

User enter ID and Pass into login. User's data is in the database. User can log into the system.



Alternative course of event 1:

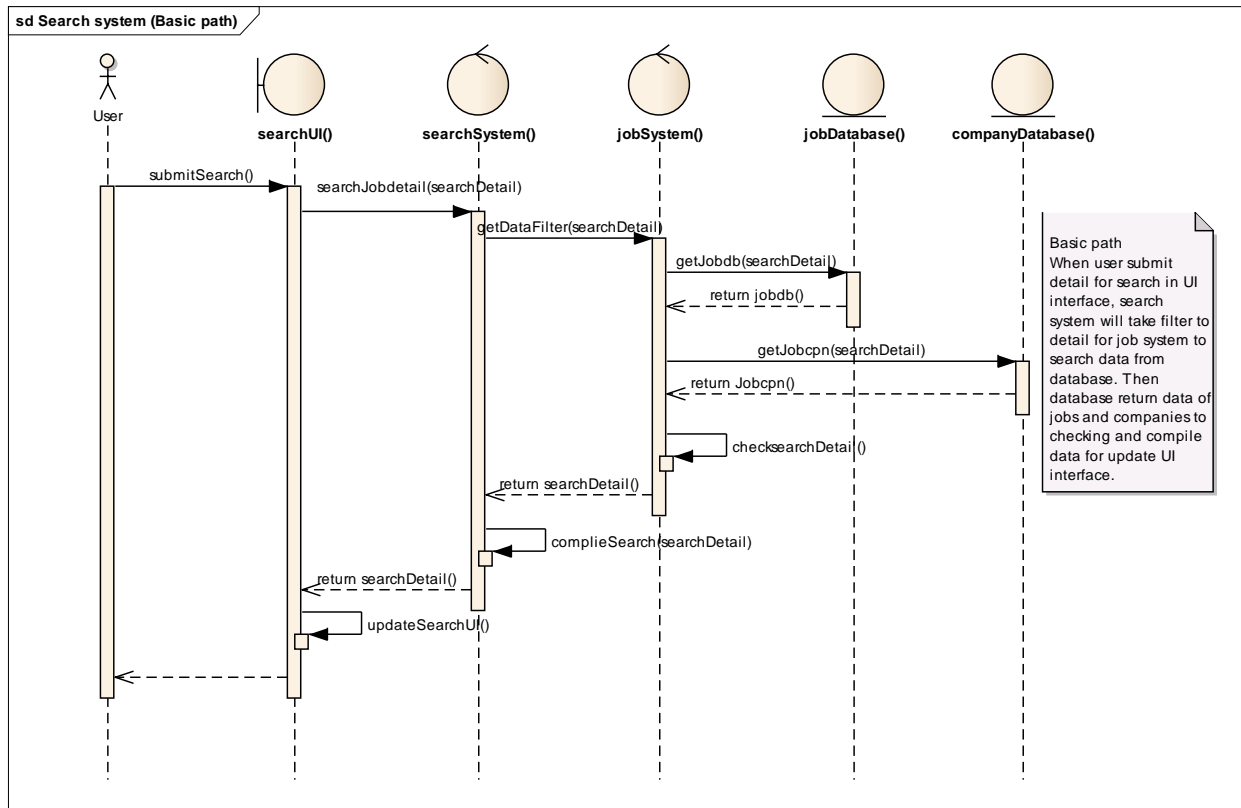
Enter the ID or Pass into the wrong login. Send ID and Pass to check in the database that the ID and Pass information is correct in the data in the database and then return the value False.



Alternative course of event 2:

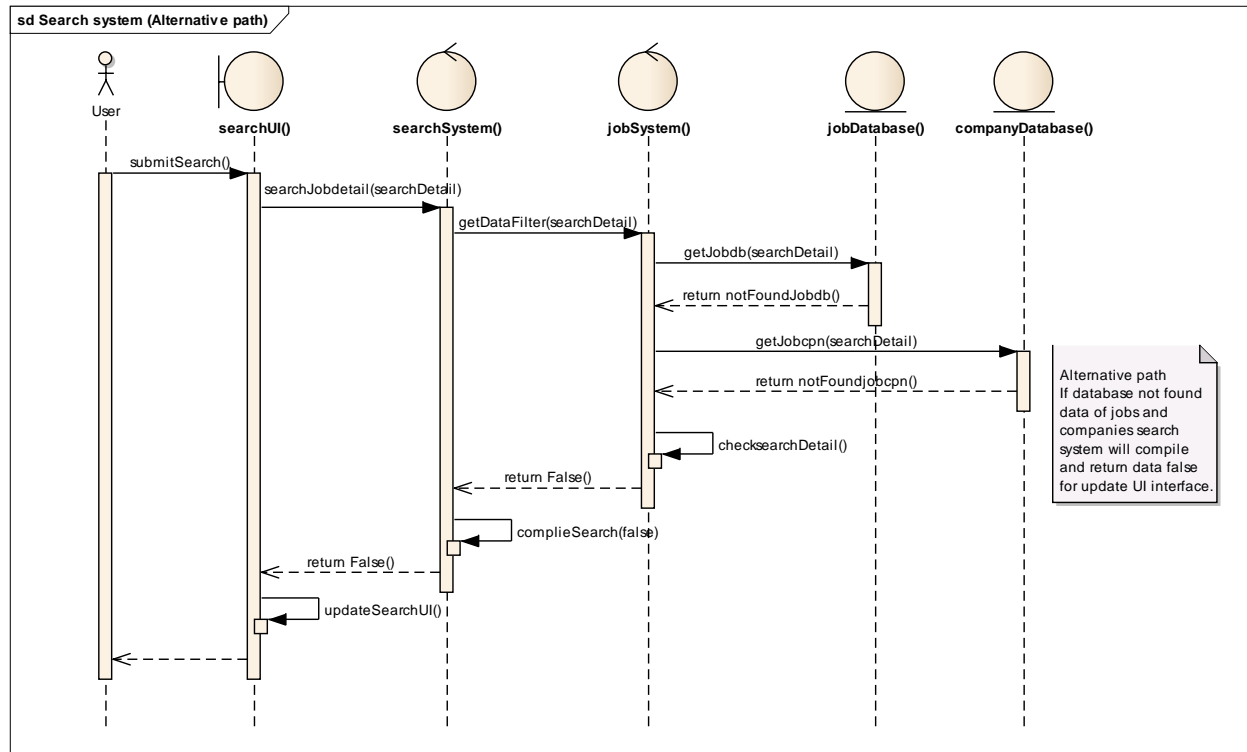
Enter the ID or Pass into the login, pass the ID and Pass to check the database that the ID and Pass information in the database and then return to False value.

Search



Basic course of event:

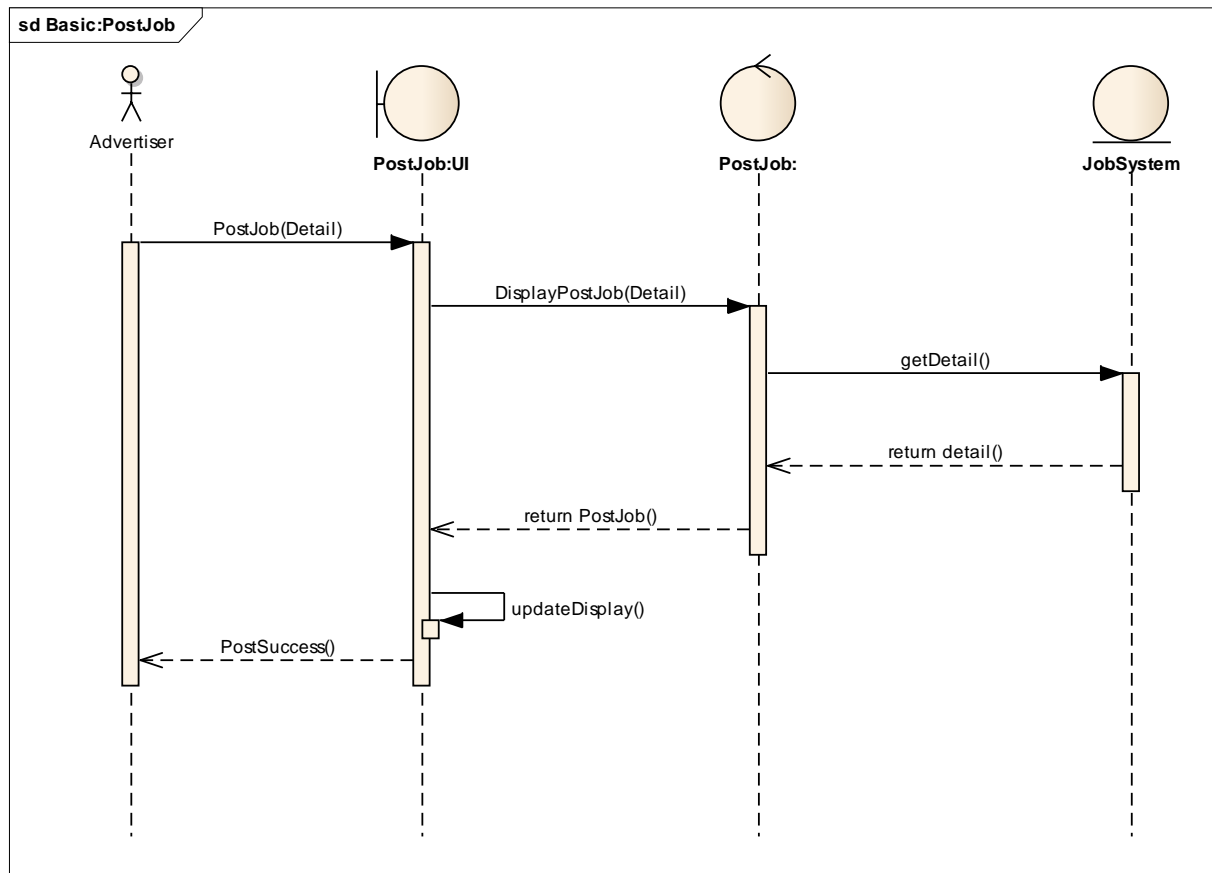
When user submit detail for search in UI interface, search system will take filter to detail for job system to search data from database. Then database return data of jobs and companies to checking and compile data for update UI interface.



Alternative course of event:

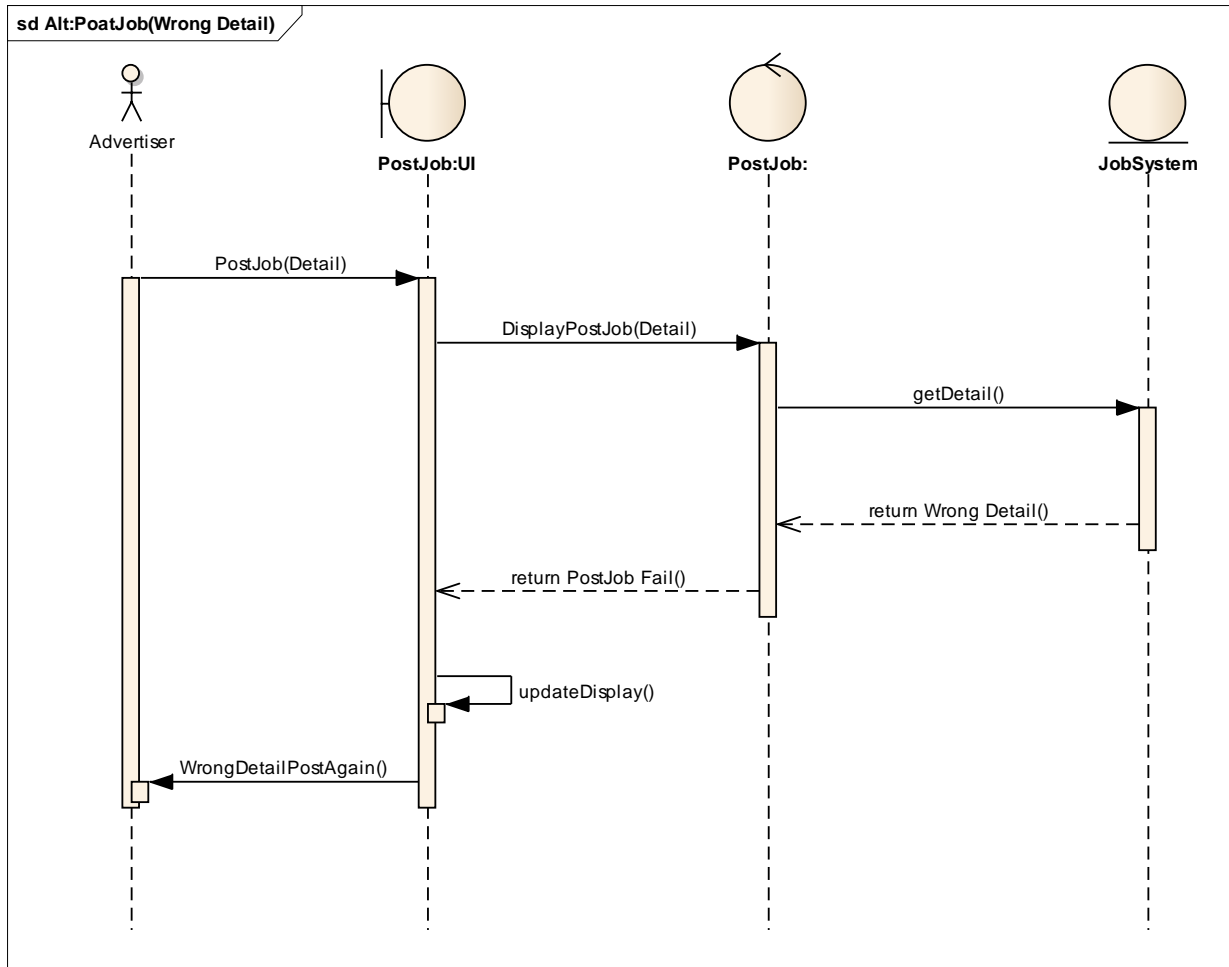
If database not found data of jobs and companies search system will compile and return data false for update UI interface.

PostJob



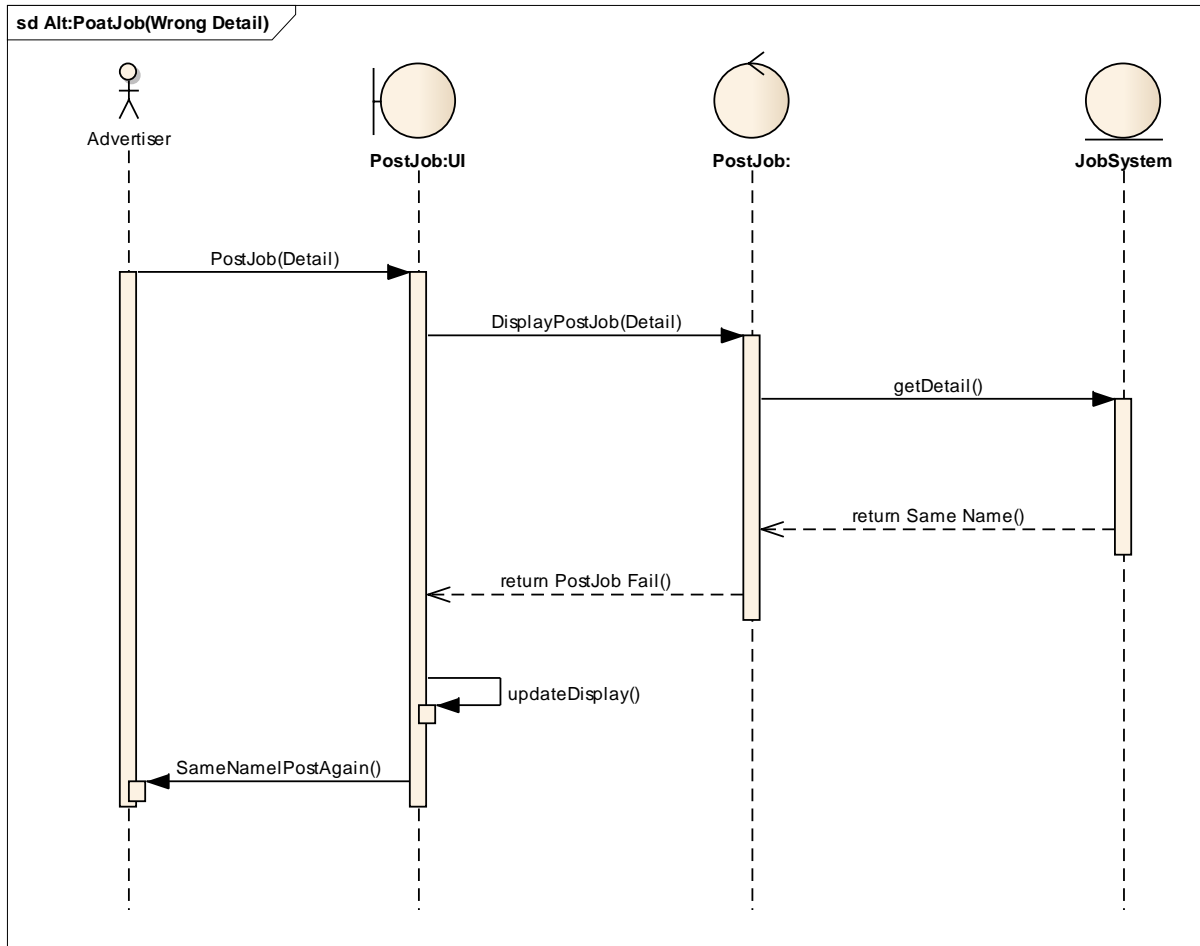
Basic Course of Event : PostJob

Advertiser input detail for Postjob and system request display the post of detail and JobSystem get Detail and check PostJobName and detail if PostJobName is unique and detail is right return detail and return display PostJob and update display and show Advertiser PostSucces.



Alternative Course of Event : PostJob with wrong detail

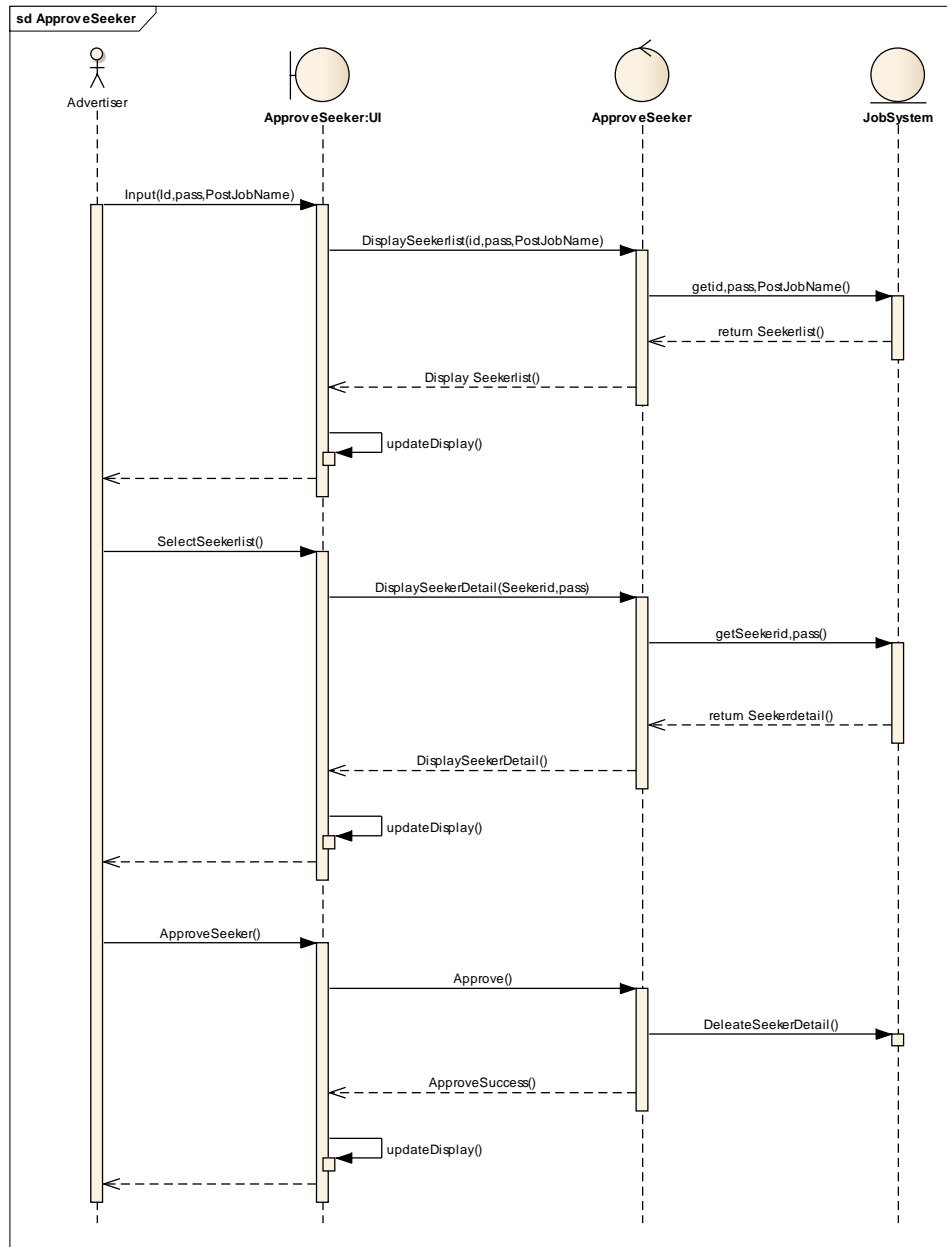
Advertiser input detail for Postjob and system request display the post of detail and JobSystem get Detail and check PostJobName and detail if PostJobName is unique but detail is wrong return Wrong detail and return PostJob Fail and update display and show Advertiser WrongDetailPostAgain.



Alternative Course of Event : PostJob with same PostJobName

Advertiser input detail for Postjob and system request display the post of detail and JobSystem get Detail and check PostJobName and detail if PostJobName isn't unique return Same Name and return PostJob Fail and update display and show Advertiser SameNamePostAgain.

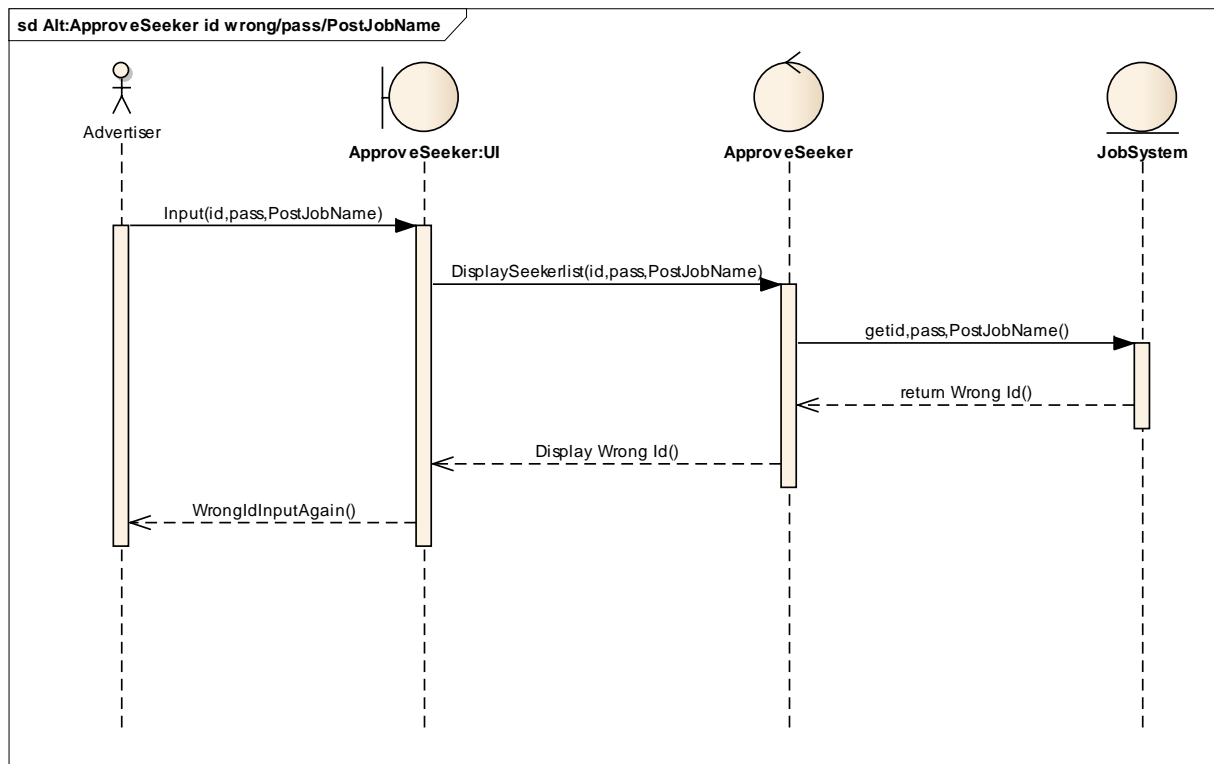
ApproveSeeker



Basic Course of Event : ApproveSeeker

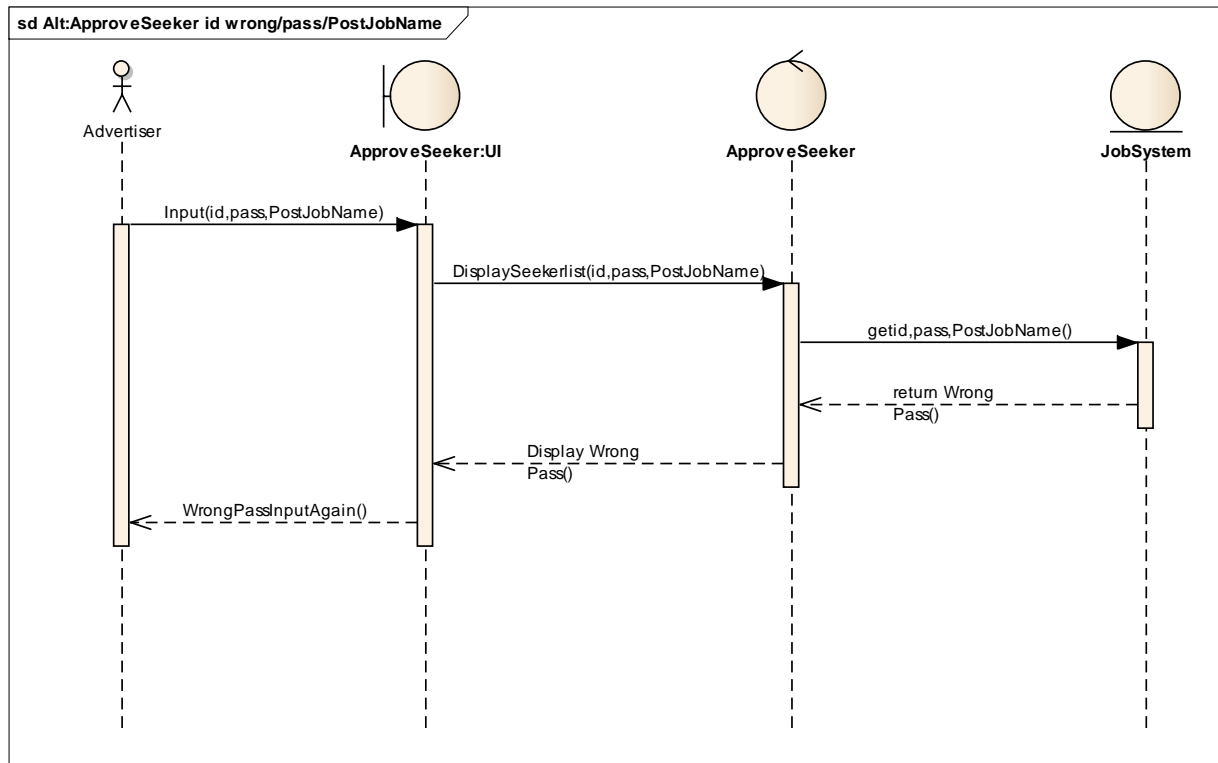
Advertiser input id,pass and PostJobName and system request display Seekerlist and JobSystem get id,pass and PostJobName and check them if all is correct return Seekerlist and display Seekerlist then select Seekerlist and system request display that SeekerDetail and JobSystem get Seekerid,pass

and return Seekerdetail to display. Finally Asvertiser ApproveSeeker for work and the JobSystem delete that Seeker detail from Seekerlist and return Approve Success.



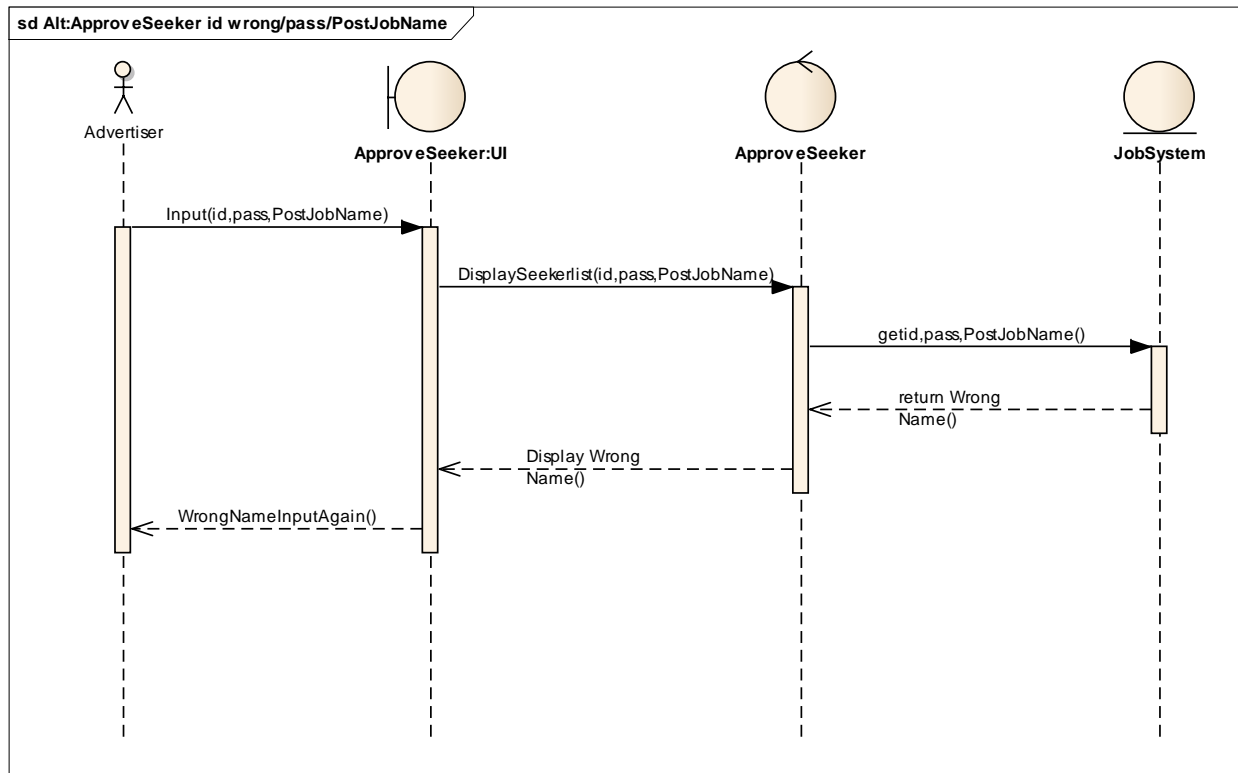
Alterbative Course of Event : ApproveSeeker Wrong Id

Advertiser input id,pass and PostJobName and system request display Seekerlist and JobSystem get id,pass and PostJobName and check them if id is incorrect return Wrong Id and display Wrong Id then show try again.



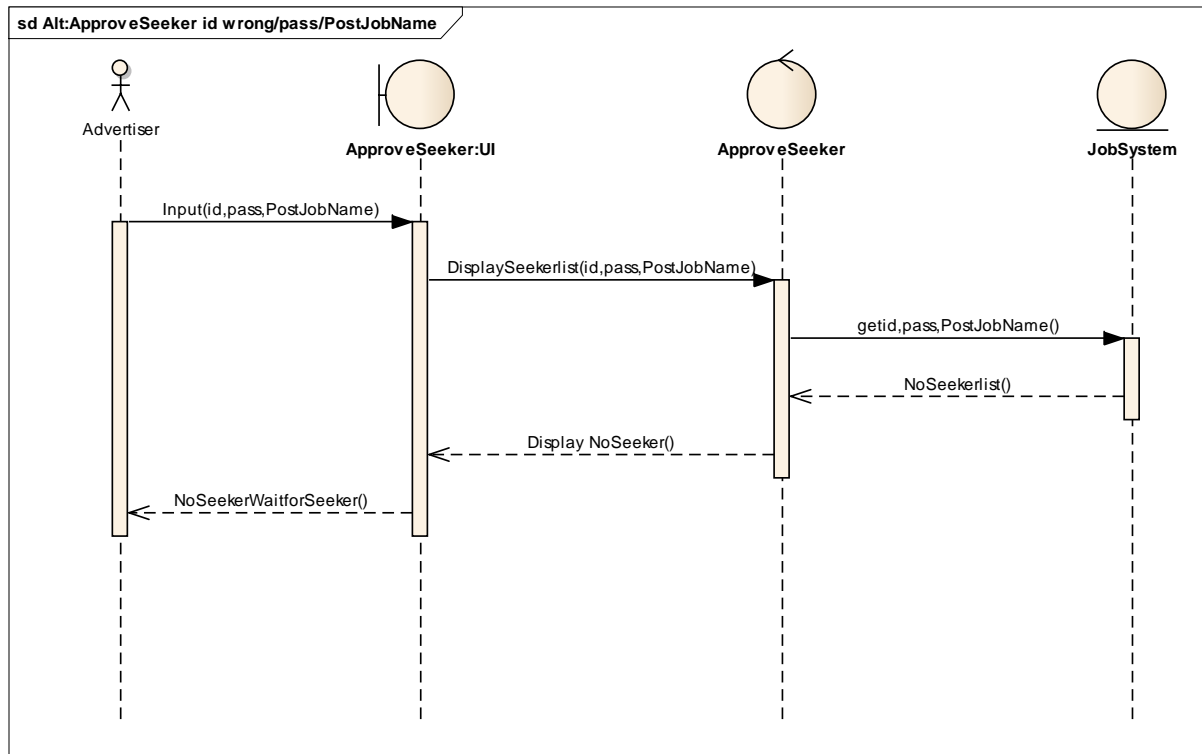
Alterbative Course of Event : ApproveSeeker Wrong Password

Advertiser input id,pass and PostJobName and system request display Seekerlist and JobSystem get id,pass and PostJobName and check them if password is incorrecr return Wrong Pass and display Wrong Pass then show try again.



Alterbative Course of Event : ApproveSeeker Wrong PostJobName

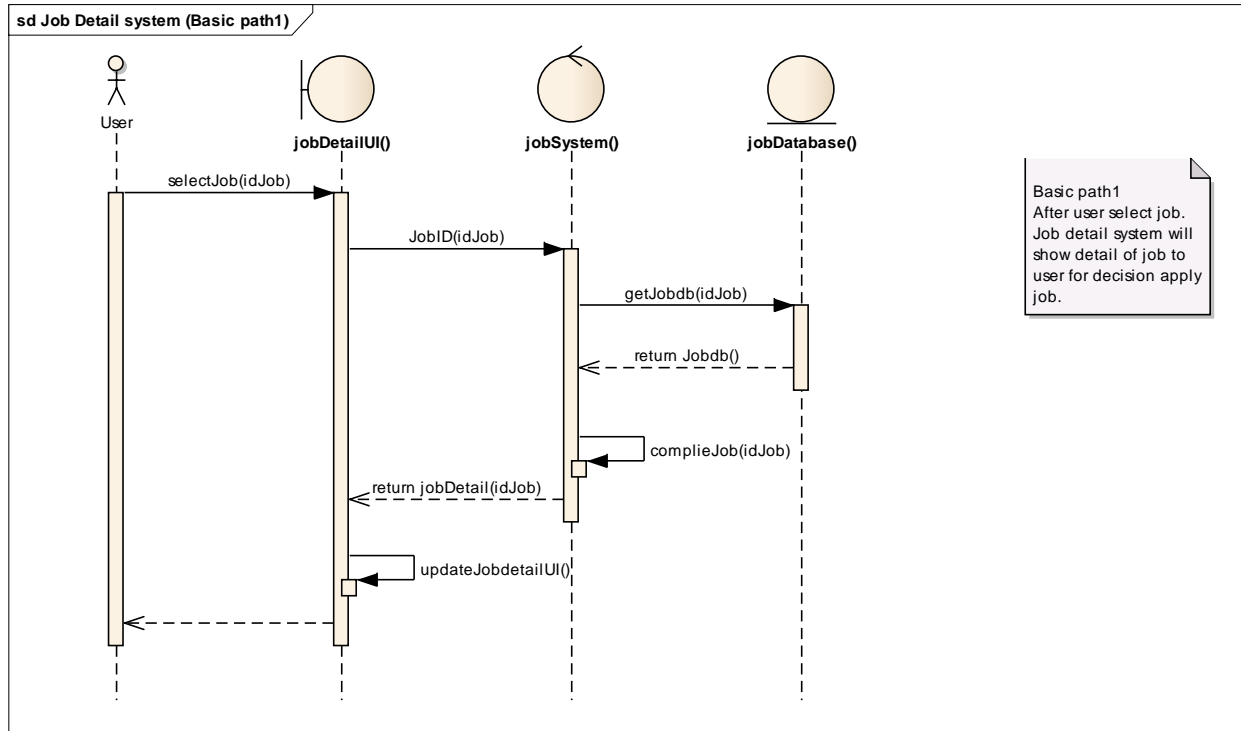
Advertiser input id,pass and PostJobName and system request display Seekerlist and JobSystem get id,pass and PostJobName and check them if PostJobName is incorrect return Wrong Name and display Wrong Name then show try again.



Alterbative Course of Event : ApproveSeeker NoSeeker

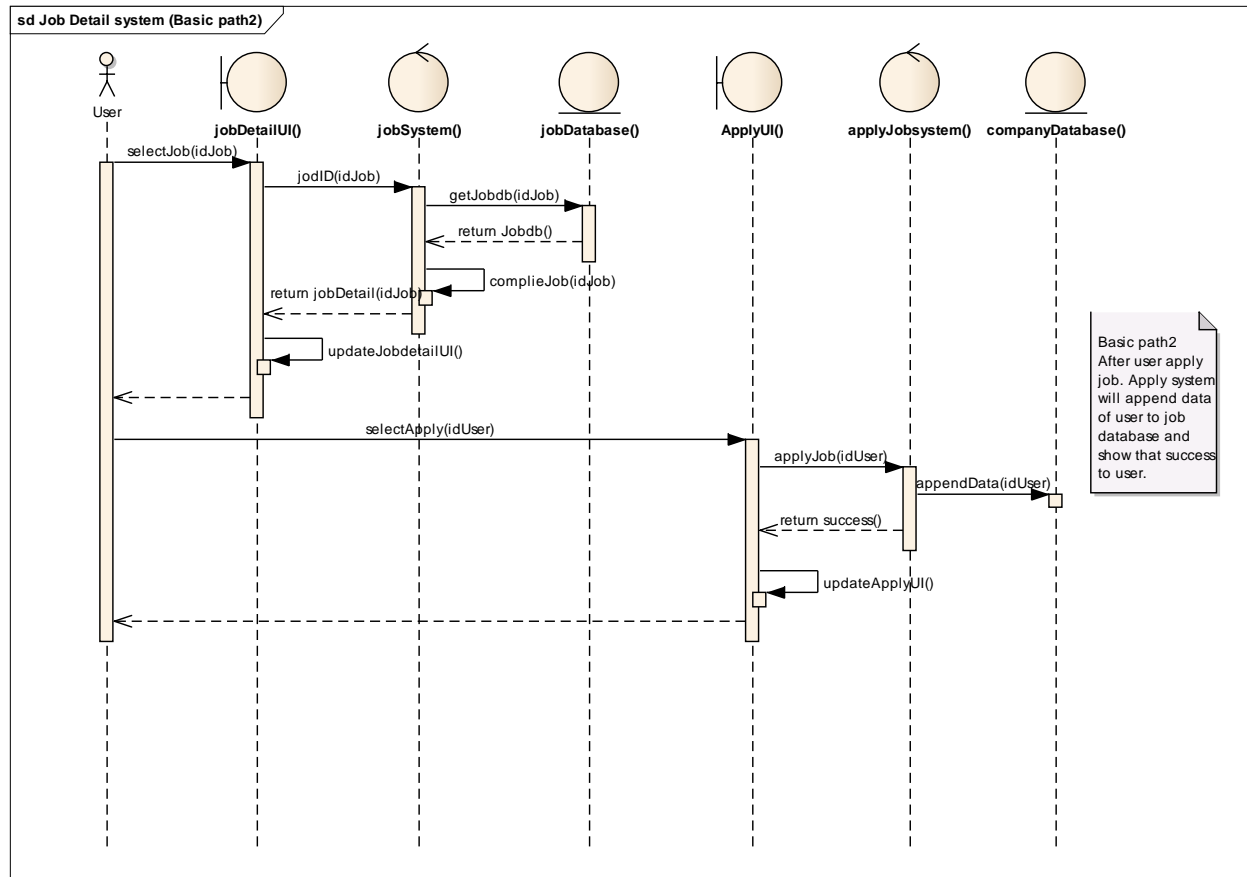
Advertiser input id,pass and PostJobName and system request display Seekerlist and JobSystem get id,pass and PostJobName and check them if all is correct but no Seekerlist return NoSeeker and display NoSeeker then show wait for Seeker.

JobDetail



Basic course of event 1:

After user select job. Job detail system will show detail of job to user for decision apply job.



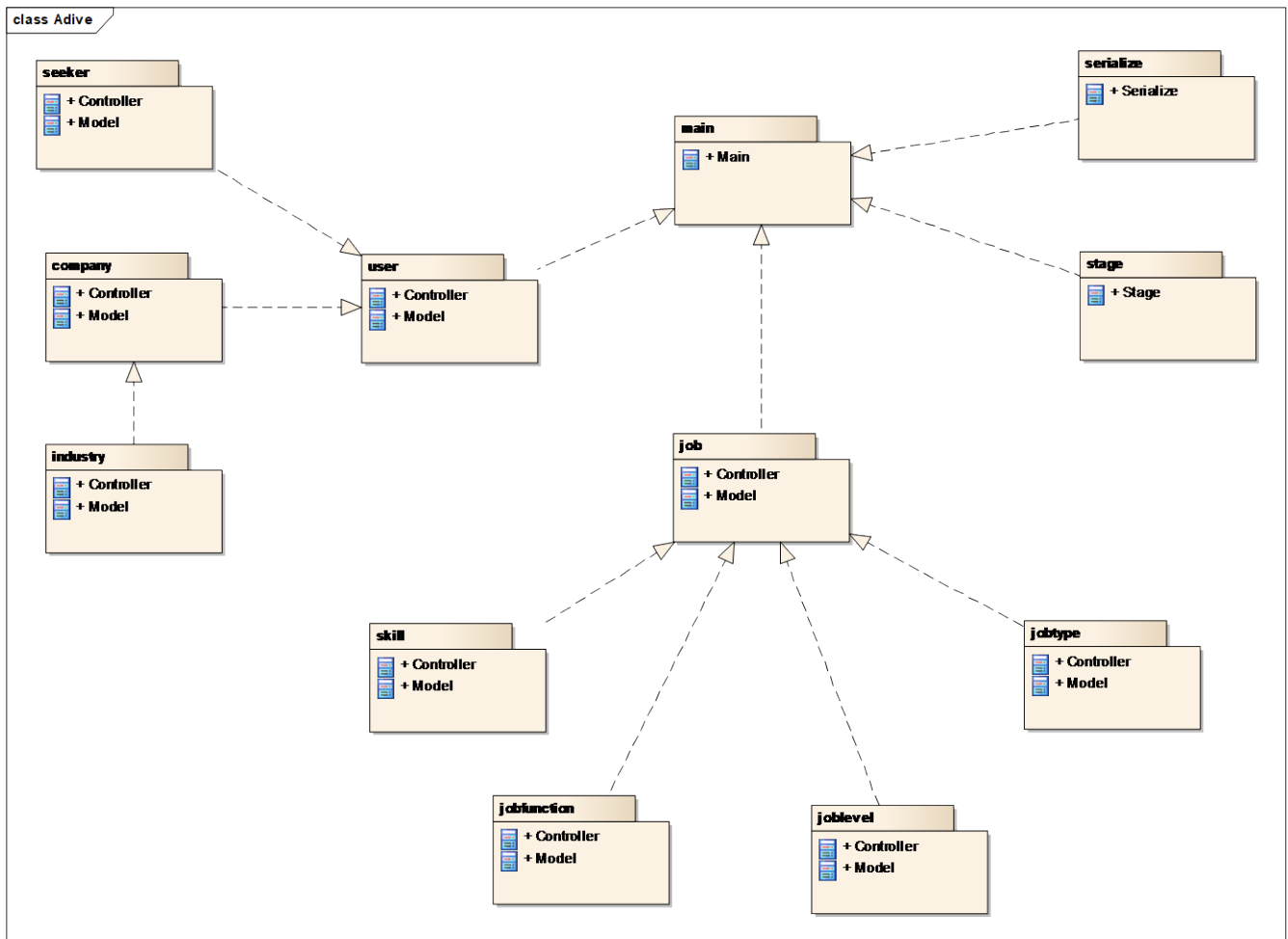
Basic course of event 2:

After user apply job. Apply system will append data of user to job database and show that success to user.

Chapter III

Design

Class diagram



class main

Main	
+	<u>main()</u> :void
	start(javafx.stage.Stage) :void

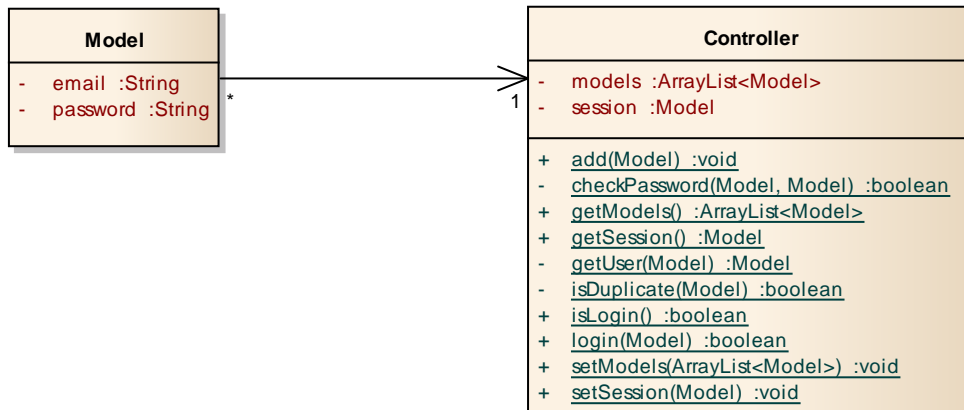
class serialize

Serialize	
-	dbpath :String
	map :<String, ArrayList>
	<u>serialize</u> :Serialize
+	objRead() :void
	objWrite() :void
	<u>Serialize()</u> :void
	serializeRead() :void
	serializeWrite(Object) :void

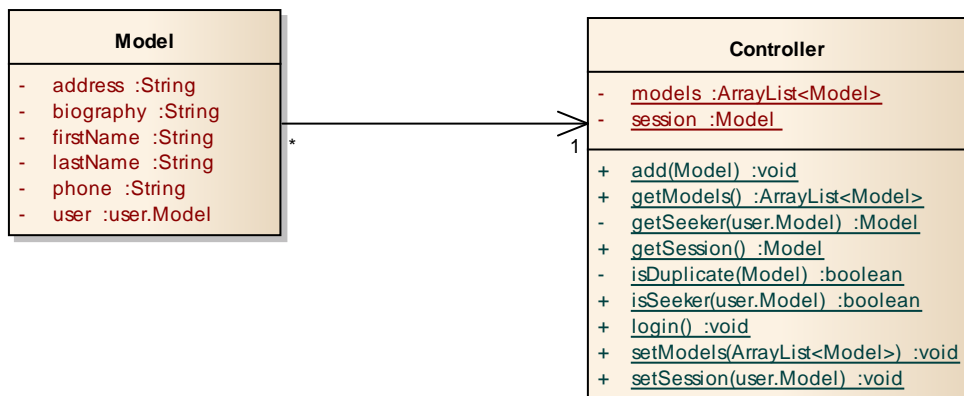
class stage

Stage	
-	primaryStage :javafx.stage.Stage
	<u>stage</u> :Stage
+	changeStage(String) :void
	getStage() :javafx.stage.Stage
	setStage(javafx.stage.Stage) :void
	setTitle(String) :void
	show() :void
	<u>Stage(javafx.stage.Stage)</u> :void

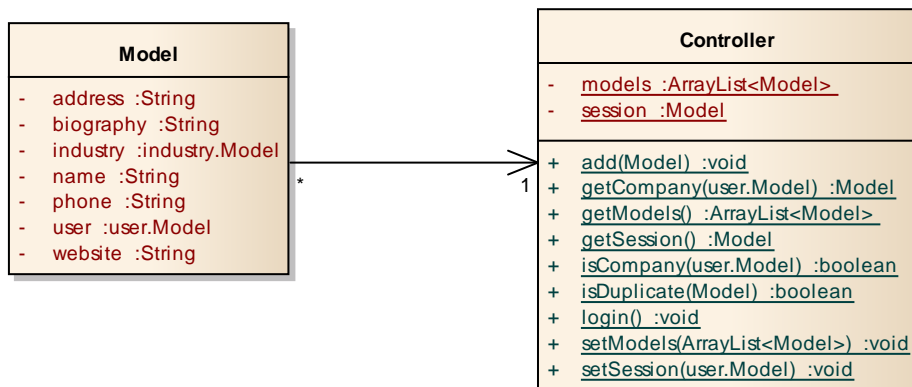
class user

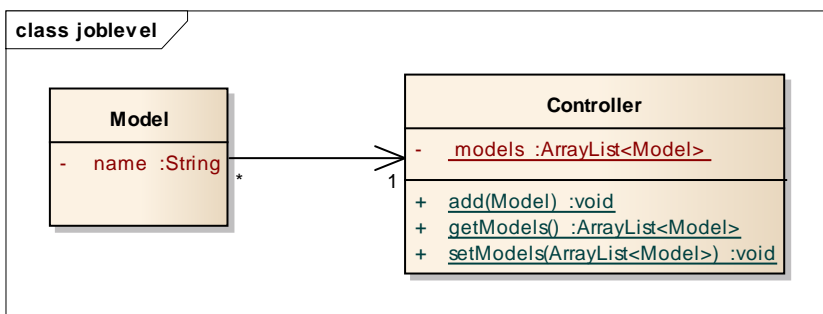
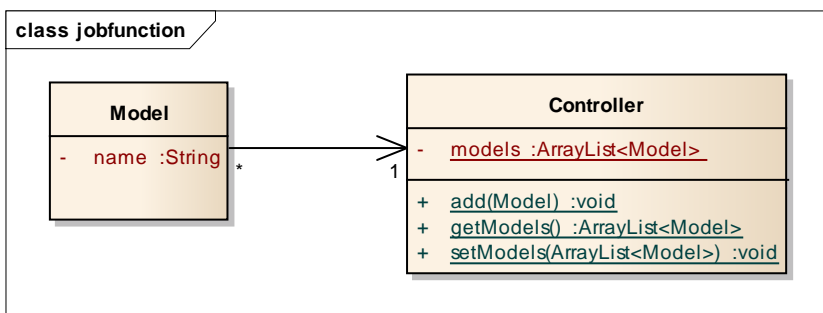
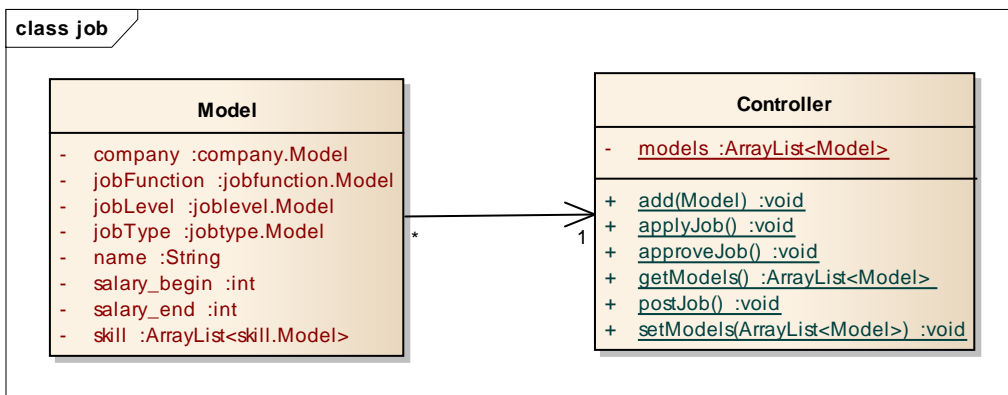
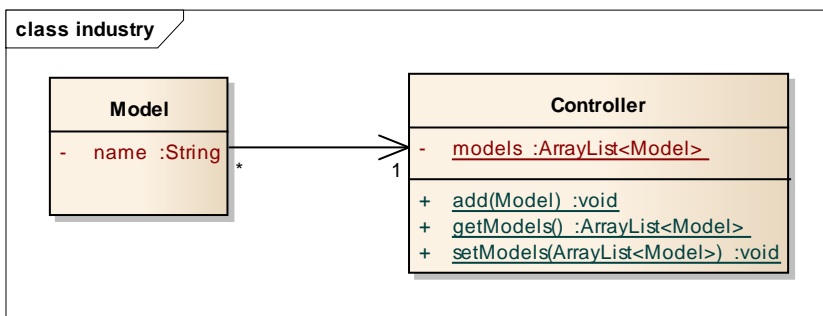


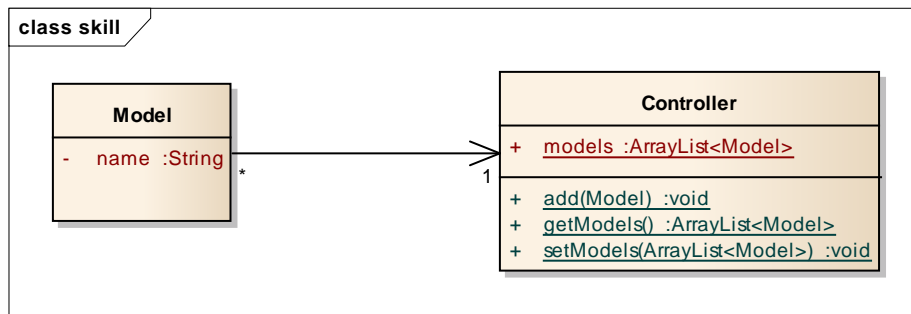
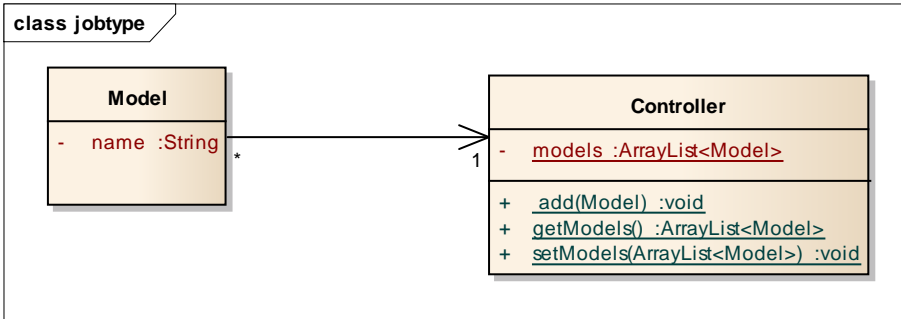
class seeker



class company

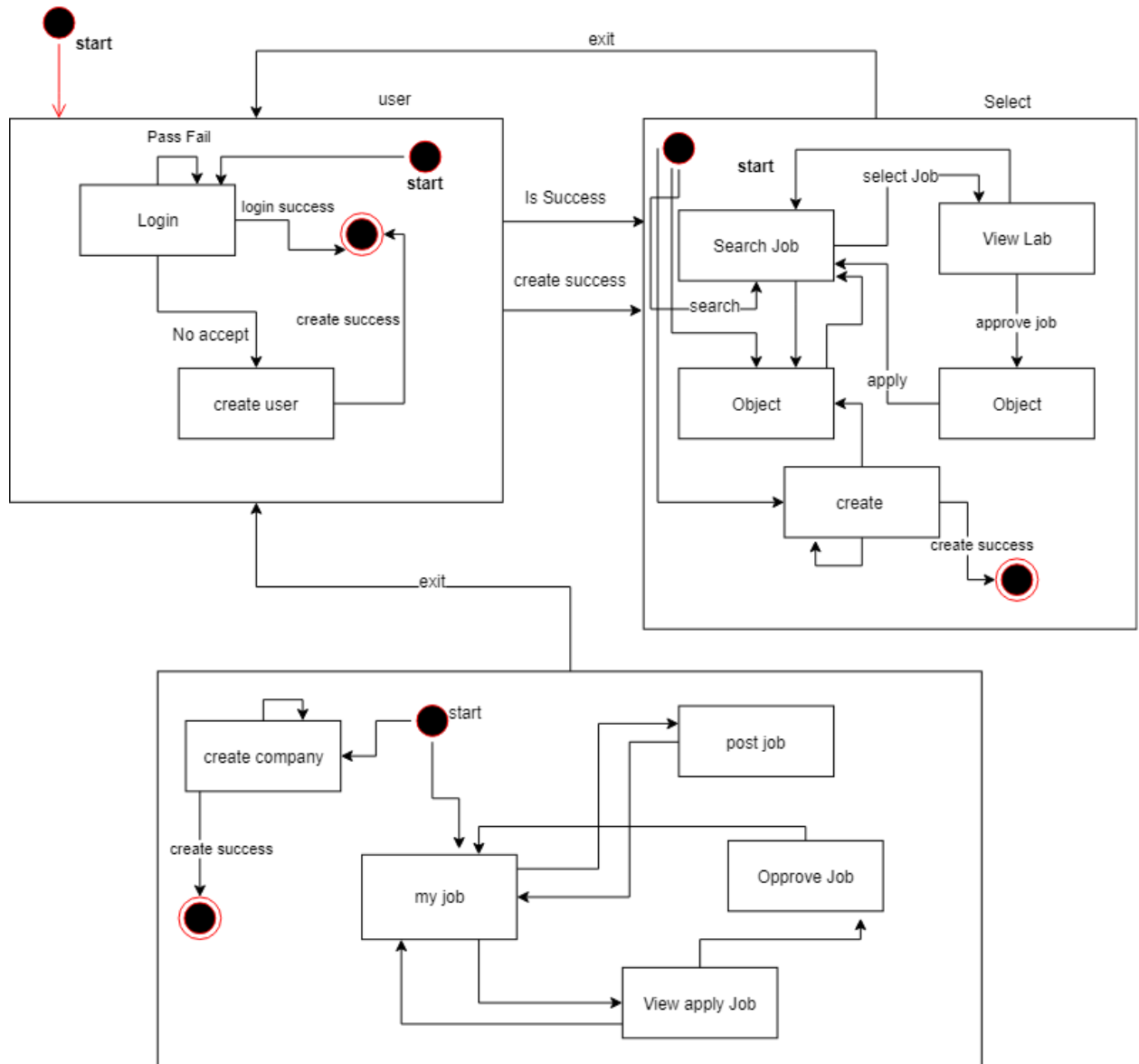






Chapter III

State-chart diagram



Chapter IV

Source Code

Source Code

Package main

Main.java

```
package main;

import javafx.application.Application;

import serialize.Serialize;

import stage.Stage;

public class Main extends Application {

    @Override

    public void start(javafx.stage.Stage primaryStage) throws Exception {

        Stage.stage(primaryStage);

        Stage.stage.changeStage("Login");

        Stage.stage.setTitle("Login");

        Stage.stage.show();

    }

    public static void main(String[] args) {
```

```
user.Controller.Controller();  
  
location.Controller.Controller();  
  
industry.Controller.Controller();  
  
company.Controller.Controller();  
  
seeker.Controller.Controller();  
  
salary.Controller.Controller();  
  
skill.Controller.Controller();  
  
jobfunction.Controller.Controller();  
  
joblevel.Controller.Controller();  
  
jobtype.Controller.Controller();  
  
job.Controller.Controller();
```

```
Serialize.Serialize();  
  
Serialize.serialize.objRead();
```

```
//    user.Test.init();  
  
//    user.Test.print();  
  
//  
  
//    company.Test.init();  
  
//    company.Test.print();  
  
//  
  
//    seeker.Test.init();
```

```
//      seeker.Test.print();

//

//      job.Test.init();

//      job.Test.print();


        launch(args);

        Serialize.serialize.objWrite();

    }

}
```

Controller.java

```
package main;

public class Controller {

}
```

Package serialize

Serialize.java

```
package serialize;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.ArrayList;
```

```
import java.util.HashMap;
```

```
public class Serialize {
```

```
    public static Serialize serialize;
```

```
    private final String dbpath = "adiv.db";
```

```
    private HashMap<String, ArrayList> map;
```

```
    public static void Serialize() {
```

```
        serialize = new Serialize();
```

```
    }
```

```
    public void objRead() {
```

```
        serializeRead();
```

```
        System.out.println("The Object  was succesfully read");
```

```
        user.Controller.controller.setModels(map.get("user"));
```

```
        location.Controller.controller.setModels(map.get("location"));
```

```
        industry.Controller.controller.setModels(map.get("industry"));
```

```
        company.Controller.controller.setModels(map.get("company"));
```

```
        seeker.Controller.controller.setModels( map.get("seeker"));
```

```
        salary.Controller.controller.setModels(map.get("salary"));
```

```
skill.Controller.controller.setModels(map.get("skill"));

jobfunction.Controller.controller.setModels(map.get("jobfunction"));

joblevel.Controller.controller.setModels(map.get("joblevel"));

jobtype.Controller.controller.setModels(map.get("jobtype"));

job.Controller.controller.setModels(map.get("job"));

}
```

```
public void objWrite() {

    HashMap<String, ArrayList> map = new HashMap<String, ArrayList>();

    map.put("user", user.Controller.controller.getModels());

    map.put("location", location.Controller.controller.getModels());

    map.put("industry", industry.Controller.controller.getModels());

    map.put("company", company.Controller.controller.getModels());

    map.put("seeker", seeker.Controller.controller.getModels());

    map.put("salary", salary.Controller.controller.getModels());

    map.put("skill", skill.Controller.controller.getModels());

    map.put("jobfunction", jobfunction.Controller.controller.getModels());

    map.put("joblevel", joblevel.Controller.controller.getModels());

    map.put("jobtype", jobtype.Controller.controller.getModels());

    map.put("job", job.Controller.controller.getModels());


    serializeWrite(map);

    System.out.println("The Object  was succesfully written to a file");

}
```



```
}
```

```
public void serializeWrite(Object serObj) {
```

```
    try {
```

```
        FileOutputStream fileOut = new FileOutputStream(dbpath);
```

```
        ObjectOutputStream objectOut = new ObjectOutputStream(fileOut);
```

```
        objectOut.writeObject(serObj);
```

```
        objectOut.close();
```

```
        fileOut.close();
```

```
    } catch (Exception ex) {
```

```
        ex.printStackTrace();
```

```
    }
```

```
}
```

```
public void serializeRead() {
```

```
    try {
```

```
        FileInputStream fileIn = new FileInputStream(dbpath);
```

```
        ObjectInputStream objectIn = new ObjectInputStream(fileIn);
```

```
        map = (HashMap<String, ArrayList>) objectIn.readObject();
```

```
        objectIn.close();
```

```
        fileIn.close();
```

```
        } catch (Exception ex) {  
            ex.printStackTrace();  
        }  
    }  
}
```

Package stage

Stage.java

```
package stage;
```

```
import javafx.fxml.FXMLLoader;
```

```
import javafx.scene.Parent;
```

```
import javafx.scene.Scene;
```

```
public class Stage {
```

```
    public static Stage stage;
```

```
    private javafx.stage.Stage primaryStage;
```

```
public static void Stage(javafx.stage.Stage primaryStage) {  
    stage = new Stage();  
    stage.setStage(primaryStage);  
    stage.getStage().setResizable(false);  
}
```

```
public void changeStage(String name) throws Exception {  
    FXMLLoader loader = new FXMLLoader(getClass().getResource("/fxml/"+name+".fxml"));  
    Parent root = loader.load();  
    Scene scene = new Scene(root);  
    primaryStage.setScene(scene);  
}
```

```
public void setTitle(String name) {  
    primaryStage.setTitle("ADIV: " + name);  
}
```

```
public void show() {  
    primaryStage.show();  
}
```

```
public javafx.stage.Stage getStage() {  
    return primaryStage;  
}
```

```
}

    public void setStage(javafx.stage.Stage primaryStage) {

        this.primaryStage = primaryStage;

    }

}
```

Package salary

Controller.java

```
package stage;

import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;

public class Stage {

    public static Stage stage;

    private javafx.stage.Stage primaryStage;

    public static void Stage(javafx.stage.Stage primaryStage) {
```

```
stage = new Stage();

stage.setStage(primaryStage);

stage.getStage().setResizable(false);

}
```

```
public void changeStage(String name) throws Exception {

    FXMLLoader loader = new FXMLLoader(getClass().getResource("/fxml/"+name+".fxml"));

    Parent root = loader.load();

    Scene scene = new Scene(root);

    primaryStage.setScene(scene);

}
```

```
public void setTitle(String name) {

    primaryStage.setTitle("ADIV: " + name);

}
```

```
public void show() {

    primaryStage.show();

}
```

```
public javafx.stage.Stage getStage() {

    return primaryStage;

}
```

```
public void setStage(javafx.stage.Stage primaryStage) {  
    this.primaryStage = primaryStage;  
}  
}
```

Model.java

```
package stage;  
  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Parent;  
import javafx.scene.Scene;  
  
public class Stage {  
  
    public static Stage stage;  
  
    private javafx.stage.Stage primaryStage;  
  
    public static void Stage(javafx.stage.Stage primaryStage) {  
        stage = new Stage();  
        stage.setStage(primaryStage);  
        stage.getStage().setResizable(false);  
    }  
}
```

```
}
```

```
public void changeStage(String name) throws Exception {
```

```
    FXMLLoader loader = new FXMLLoader(getClass().getResource("/fxml/"+name+".xml"));
```

```
    Parent root = loader.load();
```

```
    Scene scene = new Scene(root);
```

```
    primaryStage.setScene(scene);
```

```
}
```

```
public void setTitle(String name) {
```

```
    primaryStage.setTitle("ADIV: " + name);
```

```
}
```

```
public void show() {
```

```
    primaryStage.show();
```

```
}
```

```
public javafx.stage.Stage getStage() {
```

```
    return primaryStage;
```

```
}
```

```
public void setStage(javafx.stage.Stage primaryStage) {
```

```
    this.primaryStage = primaryStage;
```

```
}  
}
```

Package seeker

Controller.java

```
package seeker;
```

```
import java.util.ArrayList;
```

```
public class Controller {
```

```
    public static Controller controller;
```

```
    private static ArrayList<Model> models;
```

```
    private static Model session;
```

```
    public static void Controller() {
```

```
        controller = new Controller();
```

```
        controller.models = new ArrayList<Model>();
```

```
    }
```

```
    public void addModel(Model seeker) {
```

```
        if (hasModel(seeker)) {
```



```
        System.out.println("Error: duplicate seeker");
    } else {
        models.add(seeker);
    }
}
```

```
public Model getModel(Model seeker_input) {
    for (Model seeker : models) {
        if (seeker.equals(seeker_input)) {
            return seeker;
        }
    }

    return null;
}
```

```
public Model getModel(String firstName, String lastName) {
    for (Model seeker : models) {
        if (seeker.getFirstName().equals(firstName) && seeker.getLastName().equals(lastName)) {
            return seeker;
        }
    }
}
```

```
        return null;
    }
}
```

```
private boolean hasModel(Model seeker_input) {
    for (Model seeker : models) {
        if (seeker.equals(seeker_input)) {
            return true;
        }
    }
}
```

```
        return false;
    }
}
```

```
private Model getSeeker(user.Model user) {
    for (Model seeker : models) {
        if (seeker.getUser().getEmail().equals(user.getEmail())) {
            return seeker;
        }
    }
}
```

```
        return null;
    }
}
```

```
public boolean isSeeker(user.Model user) {  
    for (Model seeker : models) {  
        if (seeker.getUser().getEmail().equals(user.getEmail())) {  
            return true;  
        }  
    }  
  
    return false;  
}
```

```
public void syncSession() {  
    if (isSeeker(user.Controller.controller.getSession())) {  
        setSession(user.Controller.controller.getSession());  
    }  
}
```

```
public ArrayList<Model> getModels() {  
    return models;  
}
```

```
public void setModels(ArrayList<Model> models) {  
    this.models = models;  
}
```

```
public Model getSession() {  
    return session;  
}
```

```
public void setSession(user.Model user) {  
    this.session = getSeeker(user);  
}  
}
```

FXMLCreateSeeker.java

```
package seeker;  
  
import javafx.fxml.FXML;  
import javafx.fxml.Initializable;  
import javafx.scene.control.Label;  
import javafx.scene.control.TextArea;  
import javafx.scene.control.TextField;  
import javafx.scene.image.ImageView;  
import javafx.scene.input.MouseEvent;  
import stage.Stage;  
  
import java.net.URL;  
import java.util.ResourceBundle;
```

```
public class FXMLCreateSeeker implements Initializable {
```

```
    @FXML
```

```
    private TextField firstNameField;
```

```
    @FXML
```

```
    private TextField lastNameField;
```

```
    @FXML
```

```
    private TextField phoneField;
```

```
    @FXML
```

```
    private TextField educationLevelField;
```

```
    @FXML
```

```
    private TextArea addressField;
```

```
    @FXML
```

```
    private TextArea biographyField;
```

```
    @FXML
```

```
    private ImageView  nextButton;
```

```
    @FXML
```

```
    private Label label;
```

@Override

```
public void initialize(URL location, ResourceBundle resources) {
```

```
    nextButton.setPickOnBounds(true);
```

```
    nextButton.setOnMouseClicked((MouseEvent event) -> {
```

```
        if
```

```
(firstNameField.getText().trim().isEmpty())&&lastNameField.getText().trim().isEmpty()&&phoneField.g  
etText().trim().isEmpty()&&addressField.getText().trim().isEmpty()&&educationLevelField.getText().t  
rim().isEmpty() ) {
```

```
            label.setText("Please fill your detail");
```

```
        } else if (firstNameField.getText().trim().isEmpty()) {
```

```
            label.setText("Please fill your name");
```

```
        } else if (lastNameField.getText().trim().isEmpty()) {
```

```
            label.setText("Please fill your name");
```

```
        } else if (phoneField.getText().trim().isEmpty()) {
```

```
            label.setText("Please fill your telephone number");
```

```
        } else if (addressField.getText().trim().isEmpty()) {
```

```
            label.setText("Please fill your Address");
```

```
        } else if (educationLevelField.getText().trim().isEmpty()) {
```

```
            label.setText("Please fill your Education level");
```

```
        } else {
```

```

try {

    Model seeker = new
Model(user.Controller.controller.getSession(),firstNameField.getText(), lastNameField.getText(),
phoneField.getText(), addressField.getText(), educationLevelField.getText(),
biographyField.getText());

    Controller.controller.addModel(seeker);

    Controller.controller.syncSession();


    Stage.stage.changeStage("homeSeeker");

    Stage.stage.setTitle("Home Seeker");


} catch (Exception e) {

    e.printStackTrace();

}

}

});

}

}

```

FXMLHomeSeeker.java

```

package seeker;

```

```
import javafx.beans.value.ChangeListener;

import javafx.beans.value.ObservableValue;

import javafx.fxml.FXML;

import javafx.fxml.Initializable;

import javafx.scene.control.MenuButton;

import javafx.scene.control.TableColumn;

import javafx.scene.control.TableView;

import javafx.scene.control.cell.PropertyValueFactory;

import javafx.scene.image.ImageView;

import javafx.scene.input.MouseEvent;

import job.Controller;

import job.Model;

import stage.Stage;


import java.net.URL;

import java.util.ResourceBundle;
```

```
public class FXMLHomeSeeker implements Initializable {

    @FXML

    private ImageView logoutButton;

    @FXML

    private ImageView viewButton;

    @FXML
```



```
private ImageView editButton;

@FXML

private ImageView viewApplySeekerButton;

@FXML

private ImageView SeekerViewApproveButton;


@FXML

private MenuButton jobTypeField;

@FXML

private MenuButton locationField;

@FXML

private MenuButton jobLevelField;

@FXML

private MenuButton salaryField;

// @FXML

// private MenuButton skillField;

@FXML

private MenuButton jobFunctionField;


@FXML

private TableView<job.Model> table;

@FXML

private TableColumn<job.Model, String> jobNameTable;
```

@FXML

private TableColumn<job.Model, String> companyTable;

@FXML

private TableColumn<job.Model, String> jobTypeTable;

@FXML

private TableColumn<job.Model, String> jobFunctionTable;

@FXML

private TableColumn<job.Model, String> jobLevelTable;

@FXML

private TableColumn<job.Model, String> locationTable;

@FXML

private TableColumn<job.Model, String> salaryTable;

@Override

public void initialize(URL location, ResourceBundle resources) {

Controller.controller.clearFilter();

job.ControllerMenu.addJobTypeFieldTable(jobTypeField, table);

job.ControllerMenu.addJobFunctionFieldTable(jobFunctionField, table);

job.ControllerMenu.addJobLevelFieldTable(jobLevelField, table);

job.ControllerMenu.addLocationFieldTable(locationField, table);

// job.ControllerMenu.addSkillFieldTable(skillField, table);

```
job.ControllerMenu.addSalaryFieldTable(salaryField, table);

jobNameTable.setCellValueFactory(new PropertyValueFactory<>("name"));
companyTable.setCellValueFactory(new PropertyValueFactory<>("company"));
jobTypeTable.setCellValueFactory(new PropertyValueFactory<>("jobType"));
jobFunctionTable.setCellValueFactory(new PropertyValueFactory<>("jobFunction"));
jobLevelTable.setCellValueFactory(new PropertyValueFactory<>("jobLevel"));
locationTable.setCellValueFactory(new PropertyValueFactory<>("location"));
salaryTable.setCellValueFactory(new PropertyValueFactory<>("salary"));

table.setItems(job.ControllerMenu.getJobModelList());

logoutButton.setPickOnBounds(true);

logoutButton.setOnMouseClicked((MouseEvent event) -> {

    try {

        Stage.stage.changeStage("Login");

    } catch (Exception e) {

        e.printStackTrace();

    }

});

viewButton.setPickOnBounds(true);
```

```

viewButton.setOnMouseClicked((MouseEvent event) -> {

    try {

        if (table.getSelectionModel().getSelectedIndex() != -1) {

            Stage.stage.changeStage("viewJob");

        }

    } catch (Exception e) {

        e.printStackTrace();

    }

});

```

```

table.getSelectionModel().selectedItemProperty()

    .addListener(new ChangeListener<Model>() {

        @Override

        public void changed(

            ObservableValue<? extends Model> observable,

            Model oldValue, Model newValue) {

            Controller.controller.setSelect(newValue);

        }

    });

```

```
editButton.setPickOnBounds(true);

editButton.setOnMouseClicked((MouseEvent event) -> {

    try {

        Stage.stage.changeStage("profileSeeker");

    } catch (Exception e) {

        e.printStackTrace();

    }

});
```

```
viewApplySeekerButton.setPickOnBounds(true);

viewApplySeekerButton.setOnMouseClicked((MouseEvent event) -> {

    try {

        Stage.stage.changeStage("viewApplySeeker");

    } catch (Exception e) {

        e.printStackTrace();

    }

});
```

```
SeekerViewApproveButton.setPickOnBounds(true);

SeekerViewApproveButton.setOnMouseClicked((MouseEvent event) -> {
```

```
        try {  
            Stage.stage.changeStage("SeekerViewApprove");  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    });  
}  
}
```

FXMLProfileSeeker.java

```
package seeker;  
  
import javafx.fxml.FXML;  
import javafx.fxml.Initializable;  
import javafx.scene.control.TextArea;  
import javafx.scene.control.TextField;  
import javafx.scene.image.ImageView;  
import javafx.scene.input.MouseEvent;  
import stage.Stage;  
  
import java.net.URL;  
import java.util.ResourceBundle;  
  
public class FXMLProfileSeeker implements Initializable {
```

@FXML

private TextField firstNameField;

@FXML

private TextField lastNameField;

@FXML

private TextField phoneField;

@FXML

private TextField educationLevelField;

@FXML

private TextArea addressField;

@FXML

private TextArea biographyField;

@FXML

private ImageView backButton;

@FXML

private ImageView editButton;

@Override

public void initialize(URL location, ResourceBundle resources) {

Model user = Controller.controller.getSession();

firstNameField.setText(user.getFirstName());

```
lastNameField.setText(user.getLastName());

phoneField.setText(user.getPhone());

addressField.setText(user.getAddress());

educationLevelField.setText(user.getEducationLevel());

biographyField.setText(user.getBiography());


backButton.setPickOnBounds(true);

backButton.setOnMouseClicked((MouseEvent event) -> {

    try {

        Stage.stage.changeStage("HomeSeeker");

    } catch (Exception e) {

        e.printStackTrace();

    }

});

editButton.setPickOnBounds(true);
```



```
editButton.setOnMouseClicked((MouseEvent event) -> {  
  
    try {  
  
        Stage.stage.changeStage("UpdateSeeker");  
  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
  
    });  
}
```

FXMLSeekerViewApprove.java

```
package seeker;  
  
import javafx.beans.value.ChangeListener;  
import javafx.beans.value.ObservableValue;  
import javafx.collections.FXCollections;  
import javafx.collections.ObservableList;
```

```
import javafx.fxml.FXML;

import javafx.fxml.Initializable;

import javafx.scene.control.TableColumn;

import javafx.scene.control.TableView;

import javafx.scene.control.cell.PropertyValueFactory;

import javafx.scene.image.ImageView;

import javafx.scene.input.MouseEvent;

import stage.Stage;


import java.net.URL;

import java.util.ResourceBundle;


public class FXMLSeekerViewApprove implements Initializable {


    @FXML

    private TableView<job.Model> table;

    @FXML

    private TableColumn<Model, String> jobNameTable;

    @FXML

    private TableColumn<job.Model, String> companyTable;

    @FXML

    private TableColumn<job.Model, String> jobTypeTable;

    @FXML
```

```
private TableColumn<job.Model, String> jobFunctionTable;
```

```
@FXML
```

```
private TableColumn<job.Model, String> jobLevelTable;
```

```
@FXML
```

```
private TableColumn<job.Model, String> locationTable;
```

```
@FXML
```

```
private TableColumn<job.Model, String> salaryTable;
```

```
@FXML
```

```
private ImageView backButton;
```

```
@FXML
```

```
private ImageView viewButton;
```

```
private static ObservableList<job.Model> list;
```

```
@Override
```

```
public void initialize(URL location, ResourceBundle resources) {
```

```
    jobNameTable.setCellValueFactory(new PropertyValueFactory<>("name"));
```

```
    companyTable.setCellValueFactory(new PropertyValueFactory<>("company"));
```

```
    jobTypeTable.setCellValueFactory(new PropertyValueFactory<>("jobType"));
```

```
    jobFunctionTable.setCellValueFactory(new PropertyValueFactory<>("jobFunction"));
```

```
    jobLevelTable.setCellValueFactory(new PropertyValueFactory<>("jobLevel"));
```

```
locationTable.setCellValueFactory(new PropertyValueFactory<>("location"));
```

```
salaryTable.setCellValueFactory(new PropertyValueFactory<>("salary"));
```

```
table.setItems(getJobModelList());
```

```
table.getSelectionModel().selectedItemProperty()
```

```
    .addListener(new ChangeListener<job.Model>() {
```

```
        @Override
```

```
        public void changed(
```

```
            ObservableValue<? extends job.Model> observable,
```

```
            job.Model oldValue, job.Model newValue) {
```

```
                job.Controller.controller.setSelect(newValue);
```

```
            }
```

```
    });
```

```
backButton.setPickOnBounds(true);
```

```
backButton.setOnMouseClicked((MouseEvent event) -> {
```

```
    try {
```

```
        Stage.stage.changeStage("HomeSeeker");
```

```
    } catch (Exception e) {
```

```

        e.printStackTrace();
    }
});

viewButton.setPickOnBounds(true);

viewButton.setOnMouseClicked((MouseEvent event) -> {
    if (table.getSelectionModel().getSelectedIndex() != -1) {
        try {
            Stage.stage.changeStage("SeekerViewApproveDetail");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});
}

```

```

private ObservableList<job.Model> getJobModelList() {

    list = FXCollections.observableArrayList();

    seeker.Model seekerr = seeker.Controller.controller.getSession();

    for (job.Model i : job.Controller.controller.getApprove(seekerr)) {
        list.add(i);
    }
}

```

```
        return list;
    }
}
```

FXMLSeekerViewApproveDetail.java

```
package seeker;
```

```
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.Label;
import javafx.scene.control.TextArea;
import javafx.scene.control.TextField;
import javafx.scene.image.ImageView;
import javafx.scene.input.MouseEvent;
import stage.Stage;
```

```
import java.net.URL;
```

```
import java.util.ResourceBundle;
```

```
public class FXMLSeekerViewApproveDetail implements Initializable {
```

```
    @FXML
```

```
    private TextField jobNameField;
```

```
    @FXML
```

```
private TextField companyNameField;
```

```
@FXML
```

```
private TextArea detailApproveField;
```

```
@FXML
```

```
private ImageView backButton;
```

```
@FXML
```

```
private Label label;
```

```
@Override
```

```
public void initialize(URL location, ResourceBundle resources) {
```

```
    job.Model sel = job.Controller.controller.getSelect();
```

```
    jobNameField.setText(sel.getName());
```

```
    companyNameField.setText(sel.getCompany().getName());
```

```
    detailApproveField.setText(sel.getDetail());
```

```
    backButton.setPickOnBounds(true);
```

```
    backButton.setOnMouseClicked((MouseEvent event) -> {
```

```
        try {
```

```
            Stage.stage.changeStage("SeekerViewApprove");
```

```
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    });  
}  
}
```

FXMLUpdateSeeker.java

```
package seeker;  
  
import javafx.fxml.FXML;  
import javafx.fxml.Initializable;  
import javafx.scene.control.Label;  
import javafx.scene.control.PasswordField;  
import javafx.scene.control.TextArea;  
import javafx.scene.control.TextField;  
import javafx.scene.image.ImageView;  
import javafx.scene.input.MouseEvent;  
import stage.Stage;  
  
import java.net.URL;  
import java.util.ResourceBundle;  
  
public class FXMLUpdateSeeker implements Initializable {
```


@FXML

private TextField firstNameField;

@FXML

private TextField lastNameField;

@FXML

private TextField phoneField;

@FXML

private TextField educationLevelField;

@FXML

private TextField emailField;

@FXML

private PasswordField passwordField;

@FXML

private PasswordField confirmPasswordField;

@FXML

private TextArea addressField;

@FXML

private TextArea biographyField;

@FXML

private ImageView backButton;

@FXML

```
private ImageView okButton;
```

```
@FXML
```

```
private Label label;
```

```
@Override
```

```
public void initialize(URL location, ResourceBundle resources) {
```

```
    Model user = Controller.controller.getSession();
```

```
    firstNameField.setText(user.getFirstName());
```

```
    lastNameField.setText(user.getLastName());
```

```
    phoneField.setText(user.getPhone());
```

```
    addressField.setText(user.getAddress());
```

```
    educationLevelField.setText(user.getEducationLevel());
```

```
    biographyField.setText(user.getBiography());
```

```
    emailField.setText(user.getUser().getEmail());
```

```
    backButton.setPickOnBounds(true);
```

```
    backButton.setOnMouseClicked((MouseEvent event) -> {
```

```
        try {
```

```

        Stage.stage.changeStage("HomeSeeker");

    } catch (Exception e) {

        e.printStackTrace();

    }

});

okButton.setPickOnBounds(true);

okButton.setOnMouseClicked((MouseEvent event) -> {

    if (firstNameField.getText().trim().isEmpty() && lastNameField.getText().trim().isEmpty() &&
        phoneField.getText().trim().isEmpty() && addressField.getText().trim().isEmpty() &&
        educationLevelField.getText().trim().isEmpty()) {

        label.setText("Please fill your detail");

    } else if (firstNameField.getText().trim().isEmpty()) {

        label.setText("Please fill your name");

    } else if (lastNameField.getText().trim().isEmpty()) {

        label.setText("Please fill your name");

    } else if (phoneField.getText().trim().isEmpty()) {

        label.setText("Please fill your telephone number");

    } else if (addressField.getText().trim().isEmpty()) {

        label.setText("Please fill your Address");

    } else if (educationLevelField.getText().trim().isEmpty()) {

```

```
        label.setText("Please fill your Education level");
    } else if (emailField.getText().trim().isEmpty()) {
        label.setText("Please fill your email");
    } else if (passwordField.getText().trim().isEmpty()) {
        label.setText("Please fill your password");
    } else if (confirmPasswordField.getText().trim().isEmpty()) {
        label.setText("Please fill your confirm password");
    } else if (passwordField.getText().equals(confirmPasswordField.getText()) == false) {
        label.setText("Error: password mai tong kun");
    } else {
        try {

            user.setFirstName(firstNameField.getText());
            user.setLastName(lastNameField.getText());
            user.setPhone(phoneField.getText());
            user.setAddress(addressField.getText());
            user.setEducationLevel(educationLevelField.getText());
            user.setBiography(biographyField.getText());
            user.getUser().setEmail(emailField.getText());
            user.getUser().setPassword(passwordField.getText());

            Stage.stage.changeStage("homeSeeker");

            Stage.stage.setTitle("Home Seeker");
```

```
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
  
    }  
  
});  
  
}  
  
}
```

Model.java

```
package seeker;  
  
import java.io.Serializable;  
import java.util.Objects;  
  
public class Model implements Serializable {  
  
    private user.Model user;  
  
    private String firstName;  
    private String lastName;  
    private String phone;
```

```
private String address;
```

```
private String educationLevel;
```

```
private String biography;
```

```
public Model() {
```

```
}
```

```
public Model(user.Model user, String firstName, String lastName, String phone, String address,  
String educationLevel, String biography) {
```

```
    this.user = user;
```

```
    this.firstName = firstName;
```

```
    this.lastName = lastName;
```

```
    this.phone = phone;
```

```
    this.address = address;
```

```
    this.educationLevel = educationLevel;
```

```
    this.biography = biography;
```

```
}
```

```
public user.Model getUser() {
```

```
    return user;
```

```
}
```

```
public void setUser(user.Model user) {
```

```
    this.user = user;
```

```
}
```

```
public String getFirstName() {  
    return firstName;  
}
```

```
public void setFirstName(String firstName) {  
    this.firstName = firstName;  
}
```

```
public String getLastName() {  
    return lastName;  
}
```

```
public void setLastName(String lastName) {  
    this.lastName = lastName;  
}
```

```
public String getPhone() {  
    return phone;  
}
```

```
public void setPhone(String phone) {
```

```
        this.phone = phone;
    }
```

```
    public String getAddress() {
        return address;
    }
```

```
    public void setAddress(String address) {
        this.address = address;
    }
```

```
    public String getEducationLevel() {
        return educationLevel;
    }
```

```
    public void setEducationLevel(String educationLevel) {
        this.educationLevel = educationLevel;
    }
```

```
    public String getBiography() {
        return biography;
    }
```



```
public void setBiography(String biography) {  
    this.biography = biography;  
}
```

@Override

```
public boolean equals(Object o) {  
    if (this == o) return true;  
    if (!(o instanceof Model)) return false;  
    Model model = (Model) o;  
    return Objects.equals(getFirstName(), model.getFirstName()) &&  
        Objects.equals(getLastName(), model.getLastName());  
}
```

@Override

```
public int hashCode() {  
    return Objects.hash(getFirstName(), getLastName());  
}
```

@Override

```
public String toString() {  
    return firstName + " " + lastName;  
}  
}
```

Package skill

Controller.java

```
package skill;
```

```
import java.util.ArrayList;
```

```
public class Controller {
```

```
    public static Controller controller;
```

```
    public ArrayList<Model> models;
```

```
    public static void Controller() {
```

```
        controller = new Controller();
```

```
        controller.models = new ArrayList<Model>();
```

```
    }
```

```
    public void add(Model industry) {
```

```
        models.add(industry);
```

```
    }
```

```
    public Model getModel(String text) {
```

```
        for(Model model: models) {  
            if (model.getName().equals(text)) {  
                return model;  
            }  
        }  
        return null;  
    }  
  
    public ArrayList<Model> getModels() {  
        return models;  
    }  
  
    public void setModels(ArrayList<Model> models) {  
        this.models = models;  
    }  
}
```

Model.java

```
package skill;  
  
import java.io.Serializable;  
import java.util.Objects;
```

```
public class Model implements Serializable {
```

```
    private String name;
```

```
    public Model(String name) {
```

```
        this.name = name;
```

```
    }
```

```
    public String getName() {
```

```
        return name;
```

```
    }
```

```
    public void setName(String name) {
```

```
        this.name = name;
```

```
    }
```

```
    @Override
```

```
    public boolean equals(Object o) {
```

```
        if (this == o) return true;
```

```
        if (!(o instanceof Model)) return false;
```

```
        Model model = (Model) o;
```

```
        return Objects.equals(getName(), model.getName());
```

```
    }
```

```
@Override
```

```
public int hashCode() {
```

```
    return Objects.hash(getName());
```

```
}
```

```
@Override
```

```
public String toString() {
```

```
    return name;
```

```
}
```

```
}
```

Package user

Controller.java

```
package user;
```

```
import java.util.ArrayList;
```

```
public class Controller {
```

```
    public static Controller controller;
```

```
    private ArrayList<Model> models;
```

```
    private Model session;
```

```
public static void Controller() {  
  
    controller = new Controller();  
  
    controller.models = new ArrayList<Model>();  
  
}
```

```
public void addModel(Model user) {  
  
    if (hasModel(user)) {  
  
        System.out.println("Error: duplicate email");  
  
    } else {  
  
        models.add(user);  
  
    }  
  
}
```

```
private Model getModel(Model user_input) {  
  
    for (Model user : models) {  
  
        if (user.equals(user_input)) {  
  
            return user;  
  
        }  
  
    }  
  
    return null;  
  
}
```

```
}
```

```
private boolean hasModel(Model user_input) {
```

```
    for (Model user : models) {
```

```
        if (user.equals(user_input)) {
```

```
            return true;
```

```
        }
```

```
    }
```

```
    return false;
```

```
}
```

```
public void login(Model user_input) {
```

```
    if (hasModel(user_input) && checkPassword(user_input)) {
```

```
        session = getModel(user_input);
```

```
    }
```

```
}
```

```
private boolean checkPassword(Model user_input) {
```

```
    Model user = getModel(user_input);
```

```
    return user.getPassword().equals(user_input.getPassword());
```

```
}
```

```
public boolean isLogin() {  
    return session != null;  
}  
  
public ArrayList<Model> getModels() {  
    return models;  
}  
  
public void setModels(ArrayList<Model> models) {  
    this.models = models;  
}  
  
public Model getSession() {  
    return session;  
}  
  
public void setSession(Model session) {  
    this.session = session;  
}  
}
```

FXMLCreateUser.java


```
package user;
```

```
import javafx.event.ActionEvent;
```

```
import javafx.event.EventHandler;
```

```
import javafx.fxml.FXML;
```

```
import javafx.fxml.Initializable;
```

```
import javafx.scene.control.*;
```

```
import javafx.scene.image.ImageView;
```

```
import javafx.scene.input.MouseEvent;
```

```
import stage.Stage;
```

```
import java.net.URL;
```

```
import java.util.ResourceBundle;
```

```
public class FXMLCreateUser implements Initializable {
```

```
    @FXML
```

```
    private TextField emailField;
```

```
    @FXML
```

```
    private PasswordField passwordField;
```

```
    @FXML
```

```
private PasswordField confirmPasswordField;
```

```
@FXML
```

```
private CheckBox chooseSeeker;
```

```
@FXML
```

```
private CheckBox chooseCompany;
```

```
@FXML
```

```
private ImageView backButton;
```

```
@FXML
```

```
private ImageView nextButton;
```

```
@FXML
```

```
private Label label;
```

```
@Override
```

```
public void initialize(URL location, ResourceBundle resources) {
```

```
    backButton.setPickOnBounds(true);
```

```
    backButton.setOnMouseClicked((MouseEvent event) -> {
```

```
        try {
```

```
            Stage.stage.changeStage("Login");
```

```
        } catch (Exception e) {
```

```
            e.printStackTrace();
```

```
        }
```

```
});
```

```
nextButton.setPickOnBounds(true);
```

```
nextButton.setOnMouseClicked((MouseEvent event) -> {
```

```
    try {
```

```
        if (emailField.getText().trim().isEmpty()) {
```

```
            label.setText("Error: email is empty");
```

```
        } else if (passwordField.getText().trim().isEmpty()) {
```

```
            label.setText("Error: password is empty");
```

```
        } else if (confirmPasswordField.getText().trim().isEmpty()) {
```

```
            label.setText("Error: confrime password is empty");
```

```
        } else if (passwordField.getText().equals(confirmPasswordField.getText()) == false) {
```

```
            label.setText("Error: password mai tong kun");
```

```
        } else if (chooseSeeker.isSelected() && chooseCompany.isSelected()) {
```

```
            label.setText("Error: not select");
```

```
        } else if (!chooseSeeker.isSelected() && !chooseCompany.isSelected()) {
```

```
            label.setText("Error: pess select one type");
```

```
        } else {
```

```
            Model user = new Model(emailField.getText(), passwordField.getText());
```

```
            Controller.controller.addModel(user);
```

```
Controller.controller.login(user);
```

```
if (chooseSeeker.isSelected()) {
```

```
    Stage.stage.changeStage("CreateSeeker");
```

```
    Stage.stage.setTitle("New Seeker Profile");
```

```
} else if (chooseCompany.isSelected()) {
```

```
    Stage.stage.changeStage("CreateCompany");
```

```
    Stage.stage.setTitle("New Company Profile");
```

```
}
```

```
}
```

```
} catch (Exception e) {
```

```
    e.printStackTrace();
```

```
}
```

```
});
```

```
}
```

```
}
```

FXMLLogin.java

```
package user;
```

```
import javafx.event.ActionEvent;
```

```
import javafx.event.EventHandler;

import javafx.fxml.FXML;

import javafx.fxml.Initializable;

import javafx.scene.control.*;

import javafx.scene.image.ImageView;

import javafx.scene.input.MouseEvent;

import stage.Stage;
```

```
import java.net.URL;

import java.util.ResourceBundle;
```

```
public class FXMLLogin implements Initializable {
```

```
    @FXML
```

```
    private TextField emailField;
```

```
    @FXML
```

```
    private PasswordField passwordField;
```

```
    @FXML
```

```
    private ImageView loginButton;
```

```
    @FXML
```

```
private Hyperlink createUserButton;
```

```
@FXML
```

```
private Label label;
```

```
@Override
```

```
public void initialize(URL location, ResourceBundle resources) {
```

```
    loginButton.setPickOnBounds(true);
```

```
    loginButton.setOnMouseClicked((MouseEvent event) -> {
```

```
        if (emailField.getText().trim().isEmpty()) {
```

```
            label.setText("email is empty");
```

```
        } else if (passwordField.getText().trim().isEmpty()) {
```

```
            label.setText("password is empty");
```

```
        } else {
```

```
            try {
```

```
                Model user = new Model(emailField.getText(), passwordField.getText());
```

```
                Controller.controller.login(user);
```

```
                if (Controller.controller.isLogin()) {
```

```

        if (company.Controller.controller.isCompany(Controller.controller.getSession()))
    {

        company.Controller.controller.syncSession();

        Stage.stage.changeStage("homeCompany");

        Stage.stage.setTitle("Home Company");

    } else if (seeker.Controller.controller.isSeeker(Controller.controller.getSession()))
    {

        seeker.Controller.controller.syncSession();

        Stage.stage.changeStage("homeSeeker");

        Stage.stage.setTitle("Home Seeker");

    }

    } else {

        label.setText("Error: email or password incorrect");

    }

    } catch (Exception e) {

        e.printStackTrace();

    }

    }

    });

```

```
createUserButton.setOnAction(new EventHandler<ActionEvent>() {  
  
    @Override  
  
    public void handle(ActionEvent event) {  
  
        try {  
  
            Stage.stage.changeStage("CreateUser");  
  
        } catch (Exception e) {  
  
            e.printStackTrace();  
  
        }  
  
    }  
  
});  
  
}
```

Model.java

```
package user;  
  
  
import java.io.Serializable;  
  
import java.util.Objects;  
  
  
public class Model implements Serializable {  
  
  
    private String email;  
  
    private String password;
```



```
public Model() {  
    this.email = "";  
    this.password = "";  
}
```

```
public Model(String email, String password) {  
    this.email = email;  
    this.password = password;  
}
```

```
public String getEmail() {  
    return email;  
}
```

```
public void setEmail(String email) {  
    this.email = email;  
}
```

```
public String getPassword() {  
    return password;  
}
```

```
public void setPassword(String password) {  
    this.password = password;  
}
```

@Override

```
public boolean equals(Object o) {  
    if (this == o) return true;  
    if (!(o instanceof Model)) return false;  
    Model model = (Model) o;  
    return Objects.equals(getEmail(), model.getEmail());  
}
```

@Override

```
public int hashCode() {  
    return Objects.hash(getEmail());  
}
```

@Override

```
public String toString() {  
    return email;  
}  
}
```

Package company

Controller.java

```
package company;

import java.util.ArrayList;

public class Controller {

    public static Controller controller;

    private ArrayList<Model> models;
    private Model session;

    public static void Controller() {
        controller = new Controller();
        controller.models = new ArrayList<Model>();
    }

    public void addModel(Model company) {

        if (hasModel(company)) {
            System.out.println("Error: duplicate company");
        } else {
            models.add(company);
        }
    }

    private boolean hasModel(Model company_input) {
        for (Model company : models) {
            if (company.equals(company_input)) {
                return true;
            }
        }

        return false;
    }
}
```

```
public void syncSession() {  
    if (isCompany(user.Controller.controller.getSession())) {  
        setSession(user.Controller.controller.getSession());  
    }  
}
```

```
public boolean isCompany(user.Model user_input) {  
  
    for (Model company : models) {  
        if (company.getUser().getEmail().equals(user_input.getEmail())) {  
            return true;  
        }  
    }  
  
    return false;  
}
```

```
private Model getCompany(user.Model user) {  
    for (Model company : models) {  
        if (company.getUser().getEmail().equals(user.getEmail())) {  
            return company;  
        }  
    }  
  
    return null;  
}
```

```
public ArrayList<Model> getModels() {  
    return models;  
}
```

```
public void setModels(ArrayList<Model> models) {  
    this.models = models;  
}
```

```
public Model getSession() {  
    return session;  
}  
  
public void setSession(user.Model user) {  
    this.session = getCompany(user);  
}  
}
```

[FXMLCompanyApproveDetail.java](#)

```
package company;  
  
import javafx.fxml.FXML;  
import javafx.fxml.Initializable;  
import javafx.scene.control.Label;  
import javafx.scene.control.TextArea;  
import javafx.scene.control.TextField;  
import javafx.scene.image.ImageView;  
import javafx.scene.input.MouseEvent;  
import job.Controller;  
import stage.Stage;  
  
import java.net.URL;  
import java.util.ResourceBundle;  
  
public class FXMLCompanyApproveDetail implements Initializable {  
    @FXML  
    private TextField jobNameField;  
    @FXML  
    private TextField firstNameField;  
    @FXML  
    private TextArea detailApproveField;
```

```
@FXML
```

```
private ImageView backButton;
```

```
@FXML
```

```
private ImageView okButton;
```

```
@FXML
```

```
private Label label;
```

```
@Override
```

```
public void initialize(URL location, ResourceBundle resources) {
```

```
    job.Model sel = job.Controller.controller.select();
```

```
    seeker.Model selSeeker = Controller.controller.selectSeeker();
```

```
    jobNameField.setText(sel.getName());
```

```
    firstNameField.setText(selSeeker.getFirstName() + ' ' + selSeeker.getLastName());
```

```
    backButton.setPickOnBounds(true);
```

```
    backButton.setOnMouseClicked((MouseEvent event) -> {
```

```
        try {
```

```
            Stage.stage.changeStage("viewSeekerToApprove");
```

```
        } catch (Exception e) {
```

```
            e.printStackTrace();
```

```
        }
```

```
    });
```

```
    okButton.setPickOnBounds(true);
```

```
    okButton.setOnMouseClicked((MouseEvent event) -> {
```

```
        if (detailApproveField.getText().trim().isEmpty()) {
```

```
            label.setText("pless key detail");
```

```
        } else {
```

```
            job.Controller.controller.approveJobThis(detailApproveField.getText());
```

```
            try {
```

```
                Stage.stage.changeStage("viewSeekerToApprove");
```

```

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});
}
}

```

[FXMLCompanyViewjobforRemove.java](#)

```
package company;
```

```

import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.TextArea;
import javafx.scene.control.TextField;
import javafx.scene.image.ImageView;
import javafx.scene.input.MouseEvent;
import job.Controller;
import job.Model;
import stage.Stage;

```

```

import java.net.URL;
import java.util.ResourceBundle;

```

```

public class FXMLCompanyViewjobforRemove implements Initializable {
    @FXML
    private TextField jobNameField;
    @FXML
    private TextField jobTypeField;
    @FXML
    private TextField locationField;
    @FXML
    private TextField jobLevelField;
    @FXML

```

```

private TextField salaryField;
@FXML
private TextField skillField;
@FXML
private TextArea detailField;

@FXML
private ImageView backButton;
@FXML
private ImageView delButton;

@Override
public void initialize(URL location, ResourceBundle resources) {

    Model sel = Controller.controller.getSelect();

    jobNameField.setText(sel.getName());
    jobTypeField.setText(sel.getJobType().getName());
    jobLevelField.setText(sel.getJobLevel().getName());
    locationField.setText(sel.getLocation().getName());
    salaryField.setText(sel.getSalary().getName());
    skillField.setText(sel.getSkill().toString());
    detailField.setText(sel.getDetail());

    backButton.setPickOnBounds(true);
    backButton.setOnMouseClicked((MouseEvent event) -> {

        try {
            Stage.stage.changeStage("viewPostJob");
        } catch (Exception e) {
            e.printStackTrace();
        }
    });

    delButton.setPickOnBounds(true);

```



```

delButton.setOnMouseClicked((MouseEvent event) -> {
    Controller.controller.delModelThis();
    try {
        Stage.stage.changeStage("viewPostJob");
    } catch (Exception e) {
        e.printStackTrace();
    }

});
}
}

```

[FXMLCreateCompany.java](#)

```

package company;

import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.Label;
import javafx.scene.control.TextArea;
import javafx.scene.control.TextField;
import javafx.scene.image.ImageView;
import javafx.scene.input.MouseEvent;
import stage.Stage;

import java.net.URL;
import java.util.ResourceBundle;

public class FXMLCreateCompany implements Initializable {
    @FXML
    private TextField companyNameField;
    @FXML
    private TextField phoneField;
    @FXML
    private TextField websiteField;

```

```
@FXML
```

```
private TextArea addressField;
```

```
@FXML
```

```
private TextArea biographyField;
```

```
@FXML
```

```
private ImageView okButton;
```

```
@FXML
```

```
private Label label;
```

```
@Override
```

```
public void initialize(URL url, ResourceBundle resources) {
```

```
    okButton.setPickOnBounds(true);
```

```
    okButton.setOnMouseClicked((MouseEvent event) -> {
```

```
        if (companyNameField.getText().trim().isEmpty() && phoneField.getText().trim().isEmpty()
&& addressField.getText().trim().isEmpty()) {
```

```
            label.setText("Please fill your detail");
```

```
        } else if (companyNameField.getText().trim().isEmpty()) {
```

```
            label.setText("Please fill your name");
```

```
        } else if (phoneField.getText().trim().isEmpty()) {
```

```
            label.setText("Please fill your telephone number");
```

```
        } else if (addressField.getText().trim().isEmpty()) {
```

```
            label.setText("Please fill your Address");
```

```
        } else {
```

```
            try {
```

```
                industry.Model ind = industry.Controller.controller.getModels().get(0);
```

```
                Model seeker = new Model(user.Controller.controller.getSession(), ind,
companyNameField.getText(), phoneField.getText(), addressField.getText(),
websiteField.getText(), biographyField.getText());
```

```
                Controller.controller.addModel(seeker);
```

```
                Controller.controller.syncSession();
```

```

        Stage.stage.changeStage("homeCompany");
        Stage.stage.setTitle("Home Company");

    } catch (Exception e) {
        e.printStackTrace();
    }
}
});
}
}

```

[FXMLHomeCompany.java](#)

```

package company;

import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.image.ImageView;
import javafx.scene.input.MouseEvent;
import stage.Stage;

import java.net.URL;
import java.util.ResourceBundle;

public class FXMLHomeCompany implements Initializable {

    @FXML
    private ImageView postJobButton;
    @FXML
    private ImageView approveButton;
    @FXML
    private ImageView logoutButton;
    @FXML
    private ImageView profileButton;

```

```

@Override
public void initialize(URL location, ResourceBundle resources) {
    postJobButton.setPickOnBounds(true);
    postJobButton.setOnMouseClicked((MouseEvent event) -> {

        try {
            Stage.stage.changeStage("viewPostJob");
        } catch (Exception e) {
            e.printStackTrace();
        }
    });

    logoutButton.setPickOnBounds(true);
    logoutButton.setOnMouseClicked((MouseEvent event) -> {

        try {
            Stage.stage.changeStage("Login");
        } catch (Exception e) {
            e.printStackTrace();
        }
    });

    approveButton.setPickOnBounds(true);
    approveButton.setOnMouseClicked((MouseEvent event) -> {

        try {
            Stage.stage.changeStage("viewSeekerToApprove");
        } catch (Exception e) {
            e.printStackTrace();
        }
    });

    profileButton.setPickOnBounds(true);
    profileButton.setOnMouseClicked((MouseEvent event) -> {

```

```

        try {
            Stage.stage.changeStage("profileCompany");
        } catch (Exception e) {
            e.printStackTrace();
        }
    });
}
}

```

[FXMLProfileCompany.java](#)

```

package company;

import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.TextArea;
import javafx.scene.control.TextField;
import javafx.scene.image.ImageView;
import javafx.scene.input.MouseEvent;
import stage.Stage;

import java.net.URL;
import java.util.ResourceBundle;

public class FXMLProfileCompany implements Initializable {
    @FXML
    private TextField companyNameField;
    @FXML
    private TextField websiteField;
    @FXML
    private TextField phoneField;
    @FXML
    private TextArea biographyField;
    @FXML
    private TextArea addressField;
}

```

```
@FXML
```

```
private ImageView backButton;
```

```
@FXML
```

```
private ImageView editButton;
```

```
@Override
```

```
public void initialize(URL location, ResourceBundle resources) {
```

```
    Model company = Controller.controller.getSession();
```

```
    companyNameField.setText(company.getName());
```

```
    websiteField.setText(company.getWebsite());
```

```
    phoneField.setText(company.getPhone());
```

```
    addressField.setText(company.getAddress());
```

```
    biographyField.setText(company.getBiography());
```

```
    backButton.setPickOnBounds(true);
```

```
    backButton.setOnMouseClicked((MouseEvent event) -> {
```

```
        try {
```

```
            Stage.stage.changeStage("HomeCompany");
```

```
        } catch (Exception e) {
```

```
            e.printStackTrace();
```

```
        }
```

```
    });
```

```
    editButton.setPickOnBounds(true);
```

```
    editButton.setOnMouseClicked((MouseEvent event) -> {
```

```

    try {

        Stage.stage.changeStage("UpdateCompany");

    } catch (Exception e) {
        e.printStackTrace();
    }

    });
}
}

```

[FXMLUpdateCompany.java](#)

```

package company;

import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.Label;
import javafx.scene.control.PasswordField;
import javafx.scene.control.TextArea;
import javafx.scene.control.TextField;
import javafx.scene.image.ImageView;
import javafx.scene.input.MouseEvent;
import stage.Stage;

import java.net.URL;
import java.util.ResourceBundle;

public class FXMLUpdateCompany implements Initializable {
    @FXML
    private TextField companyNameField;

```

```
@FXML
private TextField websiteField;
@FXML
private TextField phoneField;
@FXML
private TextField emailField;
@FXML
private PasswordField passwordField;
@FXML
private PasswordField confirmPasswordField;
```

```
@FXML
private TextArea addressField;
@FXML
private TextArea biographyField;
```

```
@FXML
private ImageView backButton;
@FXML
private ImageView okButton;
```

```
@FXML
private Label label;
```

```
@Override
public void initialize(URL location, ResourceBundle resources) {
```

```
    Model company = Controller.controller.getSession();
```

```
    companyNameField.setText(company.getName());
    websiteField.setText(company.getWebsite());
    phoneField.setText(company.getPhone());
    addressField.setText(company.getAddress());
    biographyField.setText(company.getBiography());
    emailField.setText(company.getUser().getEmail());
```



```
backButton.setPickOnBounds(true);
backButton.setOnMouseClicked((MouseEvent event) -> {
```

```
    try {
        Stage.stage.changeStage("HomeCompany");
    } catch (Exception e) {
        e.printStackTrace();
    }
});
```

```
okButton.setPickOnBounds(true);
okButton.setOnMouseClicked((MouseEvent event) -> {
    if (companyNameField.getText().trim().isEmpty() & phoneField.getText().trim().isEmpty()
    && addressField.getText().trim().isEmpty()) {
        label.setText("Please fill your detail");
    } else if (companyNameField.getText().trim().isEmpty()) {
        label.setText("Please fill your company name");
    } else if (phoneField.getText().trim().isEmpty()) {
        label.setText("Please fill your telephone number");
    } else if (addressField.getText().trim().isEmpty()) {
        label.setText("Please fill your Address");
    } else if (emailField.getText().trim().isEmpty()) {
        label.setText("Please fill your email");
    } else if (passwordField.getText().trim().isEmpty()) {
        label.setText("Please fill your password");
    } else if (confirmPasswordField.getText().trim().isEmpty()) {
        label.setText("Please fill your confirm password");
    } else if (passwordField.getText().equals(confirmPasswordField.getText()) == false) {
        label.setText("Error: password mai tong kun");
    } else {
        try {

            company.setName(companyNameField.getText());
            company.setWebsite(websiteField.getText());
```

```
company.setPhone(phoneField.getText());
company.setAddress(addressField.getText());
company.setBiography(biographyField.getText());
company.getUser().setEmail(emailField.getText());
company.getUser().setPassword(passwordField.getText());
```

```
Stage.stage.changeStage("homeCompany");
Stage.stage.setTitle("Home Company");
```

```
    } catch (Exception e) {
        e.printStackTrace();
    }
```

```
    }
});
```

```
    }
}
```

[Model.java](#)

```
package company;
```

```
import java.io.Serializable;
import java.util.ArrayList;
import java.util.Objects;
```

```
public class Model implements Serializable {
```

```
    user.Model user;
    industry.Model industry;
```

```
    String name;
    String phone;
    String address;
    String website;
    String biography;
```

```
public Model(user.Model user, industry.Model industry, String name, String phone, String
address, String website, String biography) {
    this.user = user;
    this.industry = industry;
    this.name = name;
    this.phone = phone;
    this.address = address;
    this.website = website;
    this.biography = biography;
}
```

```
public user.Model getUser() {
    return user;
}
```

```
public void setUser(user.Model user) {
    this.user = user;
}
```

```
public industry.Model getIndustry() {
    return industry;
}
```

```
public void setIndustry(industry.Model industry) {
    this.industry = industry;
}
```

```
public String getName() {
    return name;
}
```

```
public void setName(String name) {
    this.name = name;
}
```

```
public String getPhone() {  
    return phone;  
}
```

```
public void setPhone(String phone) {  
    this.phone = phone;  
}
```

```
public String getAddress() {  
    return address;  
}
```

```
public void setAddress(String address) {  
    this.address = address;  
}
```

```
public String getWebsite() {  
    return website;  
}
```

```
public void setWebsite(String website) {  
    this.website = website;  
}
```

```
public String getBiography() {  
    return biography;  
}
```

```
public void setBiography(String biography) {  
    this.biography = biography;  
}
```

```
@Override  
public boolean equals(Object o) {
```

```
    if (this == o) return true;
    if (!(o instanceof Model)) return false;
    Model model = (Model) o;
    return Objects.equals(getName(), model.getName());
}
```

```
@Override
public int hashCode() {
    return Objects.hash(getName());
}
```

```
@Override
public String toString() {
    return name;
}
}
```

Package [industry](#)

[Controller.java](#)

```
package industry;

import java.util.ArrayList;

public class Controller {

    public static Controller controller;

    private ArrayList<Model> models;

    public static void Controller() {
        controller = new Controller();
        controller.models = new ArrayList<Model>();
    }
}
```

```
public void add(Model industry) {  
    models.add(industry);  
}  
  
public ArrayList<Model> getModels() {  
    return models;  
}  
  
public void setModels(ArrayList<Model> models) {  
    this.models = models;  
}  
}
```

[Model.java](#)

```
package industry;  
  
import java.io.Serializable;  
import java.util.Objects;  
  
public class Model implements Serializable {  
  
    private String name;  
  
    public Model(String name) {  
        this.name = name;  
    }  
  
    public String getName() {  
        return name;  
    }  
  
    public void setName(String name) {  
        this.name = name;  
    }  
}
```

```

@Override
public boolean equals(Object o) {
    if (this == o) return true;
    if (!(o instanceof Model)) return false;
    Model model = (Model) o;
    return Objects.equals(getName(), model.getName());
}

```

```

@Override
public int hashCode() {
    return Objects.hash(getName());
}

```

```

@Override
public String toString() {
    return name;
}
}

```

Package job

Controller.java

```

package job;

import java.util.ArrayList;

public class Controller {

    public static Controller controller;

    private ArrayList<Model> models;
    private ArrayList<Model> filter;
    private Model select;
    private seeker.Model selectSeeker;
}

```

```

private jobfunction.Model jobFunctionFilter;
private joblevel.Model jobLevelFilter;
private jobtype.Model jobTypeFilter;
private location.Model locationFilter;
private salary.Model salaryFilter;
private skill.Model skillFilter;

public static void Controller() {
    controller = new Controller();
    controller.models = new ArrayList<Model>();
    controller.filter = new ArrayList<Model>();
}

public void addModel(Model job) {
    if (hasModel(job) == false) {
        models.add(job);
    }
}

public void delModel(Model job) {
    models.remove(job);
}

public void delModelThis() {
    delModel(select);
}

private boolean hasModel(Model job_input) {
    for (Model job : models) {
        if (job.equals(job_input)) {
            return true;
        }
    }
}

```



```

        return false;
    }

    public boolean hasApplyJob(Model job_input, seeker.Model seeker_input) {
        for (seeker.Model seeker : job_input.getApply()) {
            if (seeker.equals(seeker_input)) {
                return true;
            }
        }

        return false;
    }

    public boolean delApplyJob(Model job_input, seeker.Model seeker_input) {
        for (int i = 0; i < job_input.getApply().size(); ++i) {
            if (job_input.getApply().get(i).equals(seeker_input)) {
                job_input.getApply().remove(i);
                return true;
            }
        }

        return false;
    }

    public void delAllApplyJob(Model job_input) {
        job_input.getApply().clear();
    }

    public void delApplyJobThis() {
        seeker.Model seekerr = seeker.Controller.controller.getSession();
        Model job = getSelect();
        delApplyJob(job, seekerr);
    }

    public boolean applyJob(Model job_input, seeker.Model seeker_input) {

```

```

        if (hasApplyJob(job_input, seeker_input)) {
            System.out.println("Error: duplicate apply job");
            return false;
        }

        job_input.getApply().add(seeker_input);
        return true;
    }

    public boolean applyJobThis() {
        seeker.Model seekerr = seeker.Controller.controller.getSession();
        Model job = getSelect();
        return applyJob(job, seekerr);
    }

    public ArrayList<Model> getApplyJob(seeker.Model seeker_input) {
        ArrayList<Model> applyJobs = new ArrayList<Model>();

        for (Model job : models) {
            if (job.getApply().contains(seeker_input)) {
                applyJobs.add(job);
            }
        }

        return applyJobs;
    }

    public ArrayList<Model> getApplyJob(company.Model company_input) {
        ArrayList<Model> applyJobs = new ArrayList<Model>();

        for (Model job : models) {
            if (job.getCompany().equals(company_input) && job.getApply().size() != 0) {
                applyJobs.add(job);
            }
        }
    }

```

```

        return applyJobs;
    }

    public void approveJob(Model job_input, seeker.Model seeker_input, String detail_input) {
        if (hasApplyJob(job_input, seeker_input)) {
//            delApplyJob(job_input, seeker_input);
            delAllApplyJob(job_input);
            job_input.setApprove(seeker_input);
            job_input.setDetail(detail_input);
        } else {
            System.out.println("Error: user not yet apply");
        }
    }

    public void approveJobThis(String detail_input) {
        Model job_input = select;
        seeker.Model seeker_input = selectSeeker;
        approveJob(job_input, seeker_input, detail_input);
    }

    public ArrayList<Model> getApprove(seeker.Model seeker_input) {
        ArrayList<Model> approveJob = new ArrayList<Model>();

        for (Model job : models) {
            if (job.getApprove().equals(seeker_input)) {
                approveJob.add(job);
            }
        }

        return approveJob;
    }

    public void filter() {

```

```

filter = new ArrayList<>(models);

for (int i = 0; i < filter.size(); ++i) {
    if (jobFunctionFilter != null)
        if (!filter.get(i).getJobFunction().equals(jobFunctionFilter)) {
            filter.remove(i);
            continue;
        }

    if (jobLevelFilter != null)
        if (!filter.get(i).getJobLevel().equals(jobLevelFilter)) {
            filter.remove(i);
            continue;
        }

    if (jobTypeFilter != null)
        if (!filter.get(i).getJobType().equals(jobTypeFilter)) {
            filter.remove(i);
            continue;
        }

    if (locationFilter != null)
        if (!filter.get(i).getLocation().equals(locationFilter)) {
            filter.remove(i);
            continue;
        }

    if (salaryFilter != null)
        if (!filter.get(i).getSalary().equals(salaryFilter)) {
            filter.remove(i);
            continue;
        }

    if (skillFilter != null)
        for (SkillModel skill : filter.get(i).getSkill())

```

```

        if (!skill.equals(skillFilter)) {
            filter.remove(i);
            continue;
        }
    }
}

```

```

public void clearFilter() {
    jobFunctionFilter = null;
    jobLevelFilter = null;
    jobTypeFilter = null;
    locationFilter = null;
    salaryFilter = null;
    skillFilter = null;
}

```

```

public ArrayList<Model> getModels(company.Model company_input) {
    ArrayList<Model> companyJob = new ArrayList<Model>();
    for (Model model : models) {
        if (model.getCompany().equals(company_input)) {
            companyJob.add(model);
        }
    }
    return companyJob;
}

```

```

public ArrayList<Model> getModels() {
    return models;
}

```

```

public void setModels(ArrayList<Model> models) {
    this.models = models;
}

```

```

public ArrayList<Model> getFilter() {

```

```
        return filter;
    }
```

```
public void setFilter(ArrayList<Model> filter) {
    this.filter = filter;
}
```

```
public Model getSelect() {
    return select;
}
```

```
public void setSelect(Model select) {
    this.select = select;
}
```

```
public seeker.Model getSelectSeeker() {
    return selectSeeker;
}
```

```
public void setSelectSeeker(seeker.Model select) {
    this.selectSeeker = select;
}
```

```
public jobfunction.Model getJobFunctionFilter() {
    return jobFunctionFilter;
}
```

```
public void setJobFunctionFilter(jobfunction.Model jobFunctionFilter) {
    this.jobFunctionFilter = jobFunctionFilter;
}
```

```
public joblevel.Model getJobLevelFilter() {
    return jobLevelFilter;
}
```

```
public void setJobLevelFilter(joblevel.Model jobLevelFilter) {  
    this.jobLevelFilter = jobLevelFilter;  
}
```

```
public jobtype.Model getJobTypeFilter() {  
    return jobTypeFilter;  
}
```

```
public void setJobTypeFilter(jobtype.Model jobTypeFilter) {  
    this.jobTypeFilter = jobTypeFilter;  
}
```

```
public location.Model getLocationFilter() {  
    return locationFilter;  
}
```

```
public void setLocationFilter(location.Model locationFilter) {  
    this.locationFilter = locationFilter;  
}
```

```
public salary.Model getSalaryFilter() {  
    return salaryFilter;  
}
```

```
public void setSalaryFilter(salary.Model salaryFilter) {  
    this.salaryFilter = salaryFilter;  
}
```

```
public skill.Model getSkilFilterl() {  
    return skilFilterl;  
}
```

```
public void setSkilFilterl(skill.Model skilFilterl) {  
    this.skilFilterl = skilFilterl;  
}
```

```
}
```

ControllerMenu.java

```
package job;
```

```
import javafx.collections.FXCollections;  
import javafx.collections.ObservableList;  
import javafx.scene.control.MenuButton;  
import javafx.scene.control.MenuItem;  
import javafx.scene.control.TableView;
```

```
public class ControllerMenu {
```

```
    private static ObservableList<job.Model> list;
```

```
    public static void addUserApprove(MenuButton userField) {  
        company.Model companyy = company.Controller.controller.getSession();  
        Model jobbb = Controller.controller.getSelect();  
        for (seeker.Model i : jobbb.getApply()) {  
            MenuItem item = new MenuItem(i.getFirstName() + ' ' + i.getLastName());  
            item.setOnAction(a -> {  
                userField.setText(i.getFirstName() + ' ' + i.getLastName());  
            });  
            userField.getItems().add(item);  
        }  
    }  
}
```

```
    public static ObservableList<Model> getJobModelList() {
```

```
        Controller.controller.filter();
```

```
        list = FXCollections.observableArrayList();  
        for (Model i : Controller.controller.getFilter()) {  
            list.add(i);  
        }  
    }  
}
```



```

    }

    return list;
}

public static void addJobTypeFieldTable(MenuButton jobTypeField, TableView<Model>
table) {
    for (jobtype.Model i : jobtype.Controller.controller.getModels()) {
        MenuItem item = new MenuItem(i.getName());
        item.setOnAction(a -> {
            jobTypeField.setText(i.getName());
            Controller.controller.setJobTypeFilter(i);
            table.setItems(getJobModelList());
        });
        jobTypeField.getItems().add(item);
    }
}

public static void addJobFunctionFieldTable(MenuButton jobFunctionField,
TableView<Model> table) {
    for (jobfunction.Model i : jobfunction.Controller.controller.getModels()) {
        MenuItem item = new MenuItem(i.getName());
        item.setOnAction(a -> {
            jobFunctionField.setText(i.getName());
            Controller.controller.setJobFunctionFilter(i);
            table.setItems(getJobModelList());
        });
        jobFunctionField.getItems().add(item);
    }
}

public static void addJobLevelFieldTable(MenuButton jobLevelField, TableView<Model>
table) {
    for (joblevel.Model i : joblevel.Controller.controller.getModels()) {
        MenuItem item = new MenuItem(i.getName());

```

```

        item.setOnAction(a -> {
            jobLevelField.setText(i.getName());
            Controller.controller.setJobLevelFilter(i);
            table.setItems(getJobModelList());
        });
        jobLevelField.getItems().add(item);
    }
}

```

```

public static void addLocationFieldTable(MenuButton locationField, TableView<Model>
table) {
    for (location.Model i : location.Controller.controller.getModels()) {
        MenuItem item = new MenuItem(i.getName());
        item.setOnAction(a -> {
            locationField.setText(i.getName());
            Controller.controller.setLocationFilter(i);
            table.setItems(getJobModelList());
        });
        locationField.getItems().add(item);
    }
}

```

```

// public static void addSkillFieldTable(MenuButton skillField, TableView<Model> table) {
//     for (skill.Model i : skill.Controller.controller.getModels()) {
//         MenuItem item = new MenuItem(i.getName());
//         item.setOnAction(a -> {
//             skillField.setText(i.getName());
//             Controller.controller.setSkillFilter(i);
//             table.setItems(getJobModelList());
//         });
//         skillField.getItems().add(item);
//     }
// }

```

```

public static void addSalaryFieldTable(MenuButton salaryField, TableView<Model> table) {

```

```

for (salary.Model i : salary.Controller.controller.getModels()) {
    MenuItem item = new MenuItem(i.getName());
    item.setOnAction(a -> {
        salaryField.setText(i.getName());
        Controller.controller.setSalaryFilter(i);
        table.setItems(getJobModelList());
    });
    salaryField.getItems().add(item);
}
}

```

```

public static void addJobTypeField(MenuButton jobTypeField) {
    for (jobtype.Model i : jobtype.Controller.controller.getModels()) {
        MenuItem item = new MenuItem(i.getName());
        item.setOnAction(a -> {
            jobTypeField.setText(i.getName());
        });
        jobTypeField.getItems().add(item);
    }
}

```

```

public static void addJobFunctionField(MenuButton jobFunctionField) {
    for (jobfunction.Model i : jobfunction.Controller.controller.getModels()) {
        MenuItem item = new MenuItem(i.getName());
        item.setOnAction(a -> {
            jobFunctionField.setText(i.getName());
        });
        jobFunctionField.getItems().add(item);
    }
}

```

```

public static void addJobLevelField(MenuButton jobLevelField) {
    for (joblevel.Model i : joblevel.Controller.controller.getModels()) {
        MenuItem item = new MenuItem(i.getName());
        item.setOnAction(a -> {

```

```

        jobLevelField.setText(i.getName());
    });
    jobLevelField.getItems().add(item);
}
}

public static void addLocationField(MenuButton locationField) {
    for (location.Model i : location.Controller.controller.getModels()) {
        MenuItem item = new MenuItem(i.getName());
        item.setOnAction(a -> {
            locationField.setText(i.getName());
        });
        locationField.getItems().add(item);
    }
}

public static void addSkillField(MenuButton skillField) {
    for (skill.Model i : skill.Controller.controller.getModels()) {
        MenuItem item = new MenuItem(i.getName());
        item.setOnAction(a -> {
            skillField.setText(i.getName());
        });
        skillField.getItems().add(item);
    }
}

public static void addSalaryField(MenuButton salaryField) {
    for (salary.Model i : salary.Controller.controller.getModels()) {
        MenuItem item = new MenuItem(i.getName());
        item.setOnAction(a -> {
            salaryField.setText(i.getName());
        });
        salaryField.getItems().add(item);
    }
}

```

```
}
```

[FXMLApproveJob.java](#)

```
package job;

import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.Button;
import javafx.scene.control.MenuButton;
import javafx.scene.control.TextArea;
import javafx.scene.control.TextField;
import javafx.scene.image.ImageView;
import javafx.scene.input.MouseEvent;
import stage.Stage;

import java.net.URL;
import java.util.ResourceBundle;

public class FXMLApproveJob implements Initializable {

    @FXML
    private TextField jobNameField;
    @FXML
    private TextField emailField;
    @FXML
    private TextField phoneField;
    @FXML
    private TextField educationLevelField;
    @FXML
    private TextArea biographyField;

    @FXML
    private MenuButton userField;
```

```
@FXML
```

```
private Button viewButton;
```

```
@FXML
```

```
private ImageView backButton;
```

```
@FXML
```

```
private ImageView okButton;
```

```
@FXML
```

```
private ImageView delButton;
```

```
@Override
```

```
public void initialize(URL location, ResourceBundle resources) {
```

```
    job.ControllerMenu.addUserApprove(userField);
```

```
    Model sel = Controller.controller.getSelect();
```

```
    jobNameField.setText(sel.getName());
```

```
    viewButton.setOnMouseClicked((MouseEvent event) -> {
```

```
        String[] parts = userField.getText().split(" ");
```

```
        seeker.Model selMenu = seeker.Controller.controller.getModel(parts[0], parts[1]);
```

```
        if (selMenu != null) {
```

```
            Controller.controller.setSelectSeeker(selMenu);
```

```
            seeker.Model selSeeker = Controller.controller.getSelectSeeker();
```

```
            emailField.setText(selSeeker.getUser().getEmail());
```

```
            phoneField.setText(selSeeker.getPhone());
```

```
            educationLevelField.setText(selSeeker.getEducationLevel());
```

```
            biographyField.setText(selSeeker.getBiography());
```

```
        }
```

```
    });
```

```
    backButton.setPickOnBounds(true);
```

```

        backButton.setOnMouseClicked((MouseEvent event) -> {

            try {
                Stage.stage.changeStage("viewSeekerToApprove");
            } catch (Exception e) {
                e.printStackTrace();
            }
        });

        okButton.setPickOnBounds(true);
        okButton.setOnMouseClicked((MouseEvent event) -> {
            if (luserField.getText().equals("select name seeker")) {
                try {
                    Stage.stage.changeStage("CompanyApproveDetail");
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });

        delButton.setPickOnBounds(true);
        delButton.setOnMouseClicked((MouseEvent event) -> {
            Controller.controller.delApplyJobThis();
            try {
                Stage.stage.changeStage("viewSeekerToApprove");
            } catch (Exception e) {
                e.printStackTrace();
            }
        });
    }
}

```

[FXMLPostJob.java](#)

package job;

```
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.Label;
import javafx.scene.control.MenuButton;
import javafx.scene.control.TextArea;
import javafx.scene.control.TextField;
import javafx.scene.image.ImageView;
import javafx.scene.input.MouseEvent;
import stage.Stage;
```

```
import java.net.URL;
import java.util.ArrayList;
import java.util.ResourceBundle;
```

```
public class FXMLPostJob implements Initializable {
```

```
    @FXML
```

```
    private TextField jobNameField;
```

```
    @FXML
```

```
    private MenuButton jobTypeField;
```

```
    @FXML
```

```
    private MenuButton salaryField;
```

```
    @FXML
```

```
    private MenuButton skillField;
```

```
    @FXML
```

```
    private MenuButton locationField;
```

```
    @FXML
```

```
    private MenuButton jobFunctionField;
```

```
    @FXML
```

```
    private MenuButton jobLevelField;
```

```
    @FXML
```

```
    private TextArea detailField;
```

```
    @FXML
```

```
    private Label label;
```



```
@FXML
```

```
private ImageView okButton;
```

```
@FXML
```

```
private ImageView backButton;
```

```
@Override
```

```
public void initialize(URL locationtion, ResourceBundle resources) {
```

```
    job.ControllerMenu.addJobTypeField(jobTypeField);
```

```
    job.ControllerMenu.addJobFunctionField(jobFunctionField);
```

```
    job.ControllerMenu.addJobLevelField(jobLevelField);
```

```
    job.ControllerMenu.addLocationField(locationField);
```

```
    job.ControllerMenu.addSkillField(skillField);
```

```
    job.ControllerMenu.addSalaryField(salaryField);
```

```
    okButton.setPickOnBounds(true);
```

```
    okButton.setOnMouseClicked((MouseEvent event) -> {
```

```
        System.out.println(jobTypeField.getText().equals("Employment type"));
```

```
        System.out.println(salaryField.getText().equals("Salary"));
```

```
        System.out.println(locationField.getText().equals("Location"));
```

```
        System.out.println(jobFunctionField.getText().equals("Job Function"));
```

```
        System.out.println(jobLevelField.getText().equals("Career level"));
```

```
        if (!jobNameField.getText().trim().isEmpty() &&
```

```
            !jobTypeField.getText().equals("Employment type") && !salaryField.getText().equals("Salary") &&
```

```
            !locationField.getText().equals("Location") && !jobFunctionField.getText().equals("Job Function")
```

```
            && !jobLevelField.getText().equals("Career level")) {
```

```
            jobfunction.Model jf =
```

```
            jobfunction.Controller.controller.getModel(jobFunctionField.getText());
```

```

        joblevel.Model jl = joblevel.Controller.controller.getModel(jobLevelField.getText());
        jobtype.Model jt = jobtype.Controller.controller.getModel(jobTypeField.getText());
        location.Model loc = location.Controller.controller.getModel(locationField.getText());
        salary.Model sa = salary.Controller.controller.getModel(salaryField.getText());

        skill.Model ski = skill.Controller.controller.getModel(skillField.getText());
        ArrayList<skill.Model> skil = new ArrayList<skill.Model>();
        skil.add(ski);

        Model jo = new job.Model(jobNameField.getText(), detailField.getText(),
company.Controller.controller.getSession(), jf, jl, jt, loc, sa, skil);
        Controller.controller.addModel(jo);

        try {
            Stage.stage.changeStage("viewPostJob");
        } catch (Exception e) {
            e.printStackTrace();
        }

    } else {

        label.setText("Please fill something to make post job perfect");

    }
});

backButton.setPickOnBounds(true);
backButton.setOnMouseClicked((MouseEvent event) -> {

    try {
        Stage.stage.changeStage("viewPostJob");
    } catch (Exception e) {
        e.printStackTrace();
    }
});

```

```

        }
    });
}
}

```

FXMLViewApplySeeker.java

```
package job;
```

```

import javafx.beans.value.ChangeListener;
import javafx.beans.value.ObservableValue;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.image.ImageView;
import javafx.scene.input.MouseEvent;
import stage.Stage;

```

```

import java.net.URL;
import java.util.ResourceBundle;

```

```

public class FXMLViewApplySeeker implements Initializable {
    @FXML
    private ImageView backButton;
    @FXML
    private ImageView delButton;
    @FXML
    private ImageView viewButton;

    @FXML
    private TableView<job.Model> table;
    @FXML
    private TableColumn<Model, String> jobNameTable;

```

```

@FXML
private TableColumn<job.Model, String> companyTable;
@FXML
private TableColumn<job.Model, String> jobTypeTable;
@FXML
private TableColumn<job.Model, String> jobFunctionTable;
@FXML
private TableColumn<job.Model, String> jobLevelTable;
@FXML
private TableColumn<job.Model, String> locationTable;
@FXML
private TableColumn<job.Model, String> salaryTable;

private ObservableList<job.Model> list;

@Override
public void initialize(URL location, ResourceBundle resources) {

    jobNameTable.setCellValueFactory(new PropertyValueFactory<>("name"));
    companyTable.setCellValueFactory(new PropertyValueFactory<>("company"));
    jobTypeTable.setCellValueFactory(new PropertyValueFactory<>("jobType"));
    jobFunctionTable.setCellValueFactory(new PropertyValueFactory<>("jobFunction"));
    jobLevelTable.setCellValueFactory(new PropertyValueFactory<>("jobLevel"));
    locationTable.setCellValueFactory(new PropertyValueFactory<>("location"));
    salaryTable.setCellValueFactory(new PropertyValueFactory<>("salary"));

    table.setItems(getJobModelList());

    table.getSelectionModel().selectedItemProperty()
        .addListener(new ChangeListener<Model>() {

            @Override
            public void changed(
                ObservableValue<? extends Model> observable,
                Model oldValue, Model newValue) {

```

```
        Controller.controller.setSelect(newValue);
    }
});
```

```
backButton.setPickOnBounds(true);
backButton.setOnMouseClicked((MouseEvent event) -> {
```

```
    try {
        Stage.stage.changeStage("homeSeeker");
    } catch (Exception e) {
        e.printStackTrace();
    }
});
```

```
viewButton.setPickOnBounds(true);
viewButton.setOnMouseClicked((MouseEvent event) -> {
```

```
    try {
        if (table.getSelectionModel().getSelectedIndex() != -1) {
            Stage.stage.changeStage("SeekerviewJobForRemove");
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
});
```

```
delButton.setPickOnBounds(true);
delButton.setOnMouseClicked((MouseEvent event) -> {
    if (table.getSelectionModel().getSelectedIndex() != -1) {
        Controller.controller.delApplyJobThis();
        try {
            Stage.stage.changeStage("viewApplySeeker");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});
```

```

        }
    }
});
}

private ObservableList<Model> getJobModelList() {

    list = FXCollections.observableArrayList();
    seeker.Model seeker1 = seeker.Controller.controller.getSession();
    for (Model i : Controller.controller.getApplyJob(seeker1)) {
        list.add(i);
    }
    return list;
}
}

```

FXMLViewApproveSeeker.java

```

package job;

import javafx.beans.value.ChangeListener;
import javafx.beans.value.ObservableValue;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.image.ImageView;
import javafx.scene.input.MouseEvent;
import stage.Stage;

import java.net.URL;
import java.util.ResourceBundle;

```

```

public class FXMLViewApproveSeeker implements Initializable {
    @FXML
    private ImageView backButton;
    @FXML
    private ImageView viewButton;

    @FXML
    private TableView<Model> table;
    @FXML
    private TableColumn<Model, String> jobNameTable;
    @FXML
    private TableColumn<Model, String> jobFunctionTable;
    @FXML
    private TableColumn<Model, String> jobTypeTable;
    @FXML
    private TableColumn<Model, String> jobLevelTable;
    @FXML
    private TableColumn<Model, String> salaryTable;

    private ObservableList<Model> list;

    @Override
    public void initialize(URL location, ResourceBundle resources) {

        jobNameTable.setCellValueFactory(new PropertyValueFactory<>("name"));
        jobFunctionTable.setCellValueFactory(new PropertyValueFactory<>("jobFunction"));
        jobTypeTable.setCellValueFactory(new PropertyValueFactory<>("jobType"));
        jobLevelTable.setCellValueFactory(new PropertyValueFactory<>("jobLevel"));
        salaryTable.setCellValueFactory(new PropertyValueFactory<>("salary"));

        table.setItems(getJobModelList());

        table.getSelectionModel().selectedItemProperty()
            .addListener(new ChangeListener<Model>() {

```

```

@Override
public void changed(
    ObservableValue<? extends Model> observable,
    Model oldValue, Model newValue) {

    Controller.controller.setSelect(newValue);
}

});

backButton.setPickOnBounds(true);
backButton.setOnMouseClicked((MouseEvent event) -> {

    try {
        Stage.stage.changeStage("homeCompany");
    } catch (Exception e) {
        e.printStackTrace();
    }
});

viewButton.setPickOnBounds(true);
viewButton.setOnMouseClicked((MouseEvent event) -> {
    if (table.getSelectionModel().getSelectedIndex() != -1) {
        try {
            Stage.stage.changeStage("ApproveJob");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});
}

private ObservableList<Model> getJobModelList() {

    list = FXCollections.observableArrayList();

```



```

        company.Model companyy = company.Controller.controller.getSession();
        for (Model i : Controller.controller.getApplyJob(companyy)) {
            list.add(i);
        }
        return list;
    }
}

```

[FXMLViewJob.java](#)

```

package job;

import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.Label;
import javafx.scene.control.TextArea;
import javafx.scene.control.TextField;
import javafx.scene.image.ImageView;
import javafx.scene.input.MouseEvent;
import stage.Stage;

import java.net.URL;
import java.util.ResourceBundle;

public class FXMLViewJob implements Initializable {
    @FXML
    private TextField jobNameField;
    @FXML
    private TextField jobTypeField;
    @FXML
    private TextField locationField;
    @FXML
    private TextField jobLevelField;
    @FXML
    private TextField salaryField;
}

```

```

@FXML
private TextField skillField;
@FXML
private TextArea detailField;

@FXML
private Label label;

@FXML
private ImageView backButton;
@FXML
private ImageView okButton;

@Override
public void initialize(URL location, ResourceBundle resources) {

    Model sel = Controller.controller.getSelect();

    jobNameField.setText(sel.getName());
    jobTypeField.setText(sel.getJobType().getName());
    jobLevelField.setText(sel.getJobLevel().getName());
    locationField.setText(sel.getLocation().getName());
    salaryField.setText(sel.getSalary().getName());
    skillField.setText(sel.getSkill().toString());
    detailField.setText(sel.getDetail());

    backButton.setPickOnBounds(true);
    backButton.setOnMouseClicked((MouseEvent event) -> {

        try {
            Stage.stage.changeStage("homeSeeker");
        } catch (Exception e) {
            e.printStackTrace();
        }
    });
}

```

```

okButton.setPickOnBounds(true);
okButton.setOnMouseClicked((MouseEvent event) -> {

    try {
        if (Controller.controller.applyJobThis()) {
            Stage.stage.changeStage("viewApplySeeker");
        } else {
            label.setText("You already apply this job!");
        }

    } catch (Exception e) {
        e.printStackTrace();
    }
});
}
}

```

[FXMLViewJobForRemove.java](#)

```

package job;

import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.TextArea;
import javafx.scene.control.TextField;
import javafx.scene.image.ImageView;
import javafx.scene.input.MouseEvent;
import stage.Stage;

import java.net.URL;
import java.util.ResourceBundle;

public class FXMLViewJobForRemove implements Initializable {
    @FXML

```

```

private TextField jobNameField;
@FXML
private TextField jobTypeField;
@FXML
private TextField locationField;
@FXML
private TextField jobLevelField;
@FXML
private TextField salaryField;
@FXML
private TextField skillField;
@FXML
private TextArea detailField;

@FXML
private ImageView backButton;
@FXML
private ImageView delButton;

@Override
public void initialize(URL location, ResourceBundle resources) {

    Model sel = Controller.controller.getSelect();

    jobNameField.setText(sel.getName());
    jobTypeField.setText(sel.getJobType().getName());
    jobLevelField.setText(sel.getJobLevel().getName());
    locationField.setText(sel.getLocation().getName());
    salaryField.setText(sel.getSalary().getName());
    skillField.setText(sel.getSkill().toString());
    detailField.setText((sel.getDetail()));

    backButton.setPickOnBounds(true);
    backButton.setOnMouseClicked((MouseEvent event) -> {

```

```

    try {

        Stage.stage.changeStage("viewApplySeeker");

    } catch (Exception e) {
        e.printStackTrace();
    }
});

delButton.setPickOnBounds(true);
delButton.setOnMouseClicked((MouseEvent event) -> {
    Controller.controller.delApplyJobThis();
    try {

        Stage.stage.changeStage("viewApplySeeker");

    } catch (Exception e) {
        e.printStackTrace();
    }
});
}
}

```

[FXMLViewPostjob.java](#)

```

package job;

import javafx.beans.value.ChangeListener;
import javafx.beans.value.ObservableValue;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.ListView;
import javafx.scene.control.TableColumn;

```

```
import javafx.scene.control.TableView;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.image.ImageView;
import javafx.scene.input.MouseEvent;
import stage.Stage;
```

```
import java.net.URL;
import java.util.ResourceBundle;
```

```
public class FXMLViewPostjob implements Initializable {
    @FXML
    private ImageView backButton;
    @FXML
    private ListView listView;
    @FXML
    private ImageView viewButton;
    @FXML
    private ImageView delButton;
    @FXML
    private ImageView postButton;

    @FXML
    private TableView<Model> table;
    @FXML
    private TableColumn<Model, String> jobNameTable;
    @FXML
    private TableColumn<job.Model, String> jobTypeTable;
    @FXML
    private TableColumn<job.Model, String> jobFunctionTable;
    @FXML
    private TableColumn<job.Model, String> jobLevelTable;
    @FXML
    private TableColumn<job.Model, String> locationTable;
    @FXML
    private TableColumn<job.Model, String> salaryTable;
```

```
@Override
```

```
public void initialize(URL location, ResourceBundle resources) {
```

```
    jobNameTable.setCellValueFactory(new PropertyValueFactory<>("name"));
    jobTypeTable.setCellValueFactory(new PropertyValueFactory<>("jobType"));
    jobFunctionTable.setCellValueFactory(new PropertyValueFactory<>("jobFunction"));
    jobLevelTable.setCellValueFactory(new PropertyValueFactory<>("jobLevel"));
    locationTable.setCellValueFactory(new PropertyValueFactory<>("location"));
    salaryTable.setCellValueFactory(new PropertyValueFactory<>("salary"));
```

```
    table.setItems(getJobModelList());
```

```
    table.getSelectionModel().selectedItemProperty()
        .addListener(new ChangeListener<Model>() {
```

```
        @Override
```

```
        public void changed(
```

```
            ObservableValue<? extends Model> observable,
            Model oldValue, Model newValue) {
```

```
                Controller.controller.setSelect(newValue);
```

```
            }
```

```
        });
```

```
    backButton.setPickOnBounds(true);
```

```
    backButton.setOnMouseClicked((MouseEvent event) -> {
```

```
        try {
```

```
            Stage.stage.changeStage("homeCompany");
```

```
        } catch (Exception e) {
```

```
            e.printStackTrace();
```

```
        }
```

```
    });
```

```

viewButton.setPickOnBounds(true);
viewButton.setOnMouseClicked((MouseEvent event) -> {
    if (table.getSelectionModel().getSelectedIndex() != -1) {
        try {
            Stage.stage.changeStage("CompanyviewJobForRemove");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});

postButton.setPickOnBounds(true);
postButton.setOnMouseClicked((MouseEvent event) -> {

    try {
        Stage.stage.changeStage("Postjob");
    } catch (Exception e) {
        e.printStackTrace();
    }
});

delButton.setPickOnBounds(true);
delButton.setOnMouseClicked((MouseEvent event) -> {
    if (table.getSelectionModel().getSelectedIndex() != -1) {
        Controller.controller.delModelThis();
        try {
            Stage.stage.changeStage("viewPostJob");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});
}

public static ObservableList<Model> getJobModelList() {

```



```

        Controller.controller.filter();

        ObservableList<job.Model> list = FXCollections.observableArrayList();
        for (Model i : Controller.controller.getModels(company.Controller.controller.getSession())) {
            list.add(i);
        }

        return list;
    }
}

```

[Model.java](#)

```

package job;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.Objects;

public class Model implements Serializable {

    private String name;
    private String detail;

    private company.Model company;
    private jobfunction.Model jobFunction;
    private joblevel.Model jobLevel;
    private jobtype.Model jobType;
    private location.Model location;
    private salary.Model salary;
    private ArrayList<skill.Model> skill;

    private ArrayList<seeker.Model> apply;
    private seeker.Model approve;
}

```

```

    public Model(String name, String detail, company.Model company, jobfunction.Model
jobFunction, joblevel.Model jobLevel, jobtype.Model jobType, location.Model location,
salary.Model salary, ArrayList<skill.Model> skill) {
        this.name = name;
        this.detail = detail;
        this.company = company;
        this.jobFunction = jobFunction;
        this.jobLevel = jobLevel;
        this.jobType = jobType;
        this.location = location;
        this.salary = salary;
        this.skill = skill;

        this.apply = new ArrayList<seeker.Model>();
        this.approve = new seeker.Model();
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getDetail() {
        return detail;
    }

    public void setDetail(String detail) {
        this.detail = detail;
    }

    public company.Model getCompany() {

```

```
        return company;
    }

    public void setCompany(company.Model company) {
        this.company = company;
    }

    public jobfunction.Model getJobFunction() {
        return jobFunction;
    }

    public void setJobFunction(jobfunction.Model jobFunction) {
        this.jobFunction = jobFunction;
    }

    public joblevel.Model getJobLevel() {
        return jobLevel;
    }

    public void setJobLevel(joblevel.Model jobLevel) {
        this.jobLevel = jobLevel;
    }

    public jobtype.Model getJobType() {
        return jobType;
    }

    public void setJobType(jobtype.Model jobType) {
        this.jobType = jobType;
    }

    public location.Model getLocation() {
        return location;
    }
}
```

```
public void setLocation(location.Model location) {  
    this.location = location;  
}
```

```
public salary.Model getSalary() {  
    return salary;  
}
```

```
public void setSalary(salary.Model salary) {  
    this.salary = salary;  
}
```

```
public ArrayList<skill.Model> getSkill() {  
    return skill;  
}
```

```
public void setSkill(ArrayList<skill.Model> skill) {  
    this.skill = skill;  
}
```

```
public ArrayList<seeker.Model> getApply() {  
    return apply;  
}
```

```
public void setApply(ArrayList<seeker.Model> apply) {  
    this.apply = apply;  
}
```

```
public seeker.Model getApprove() {  
    return approve;  
}
```

```
public void setApprove(seeker.Model approve) {  
    this.approve = approve;  
}
```

```

@Override
public boolean equals(Object o) {
    if (this == o) return true;
    if (!(o instanceof Model)) return false;
    Model model = (Model) o;
    return Objects.equals(getName(), model.getName()) &&
        Objects.equals(getCompany(), model.getCompany());
}

```

```

@Override
public int hashCode() {
    return Objects.hash(getName(), getCompany());
}

```

```

@Override
public String toString() {
    return name;
}
}

```

Package jobfunction

[Controller.java](#)

```

package jobfunction;

import java.util.ArrayList;

public class Controller {

    public static Controller controller;

    private ArrayList<Model> models;

    public static void Controller() {

```

```

        controller = new Controller();
        controller.models = new ArrayList<Model>();
    }

    public void add(Model jobFunction) {
        models.add(jobFunction);
    }

    public Model getModel(String text) {
        for(Model model: models) {
            if (model.getName().equals(text)) {
                return model;
            }
        }
        return null;
    }

    public ArrayList<Model> getModels() {
        return models;
    }

    public void setModels(ArrayList<Model> models) {
        this.models = models;
    }
}

```

[Model.java](#)

```

package jobfunction;

import java.io.Serializable;
import java.util.Objects;

public class Model implements Serializable {

```

```

private String name;

public Model(String name) {
    this.name = name;
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

@Override
public boolean equals(Object o) {
    if (this == o) return true;
    if (!(o instanceof Model)) return false;
    Model model = (Model) o;
    return Objects.equals(getName(), model.getName());
}

@Override
public int hashCode() {
    return Objects.hash(getName());
}

@Override
public String toString() {
    return name;
}
}

```

Package joblevel

Controller.java

```
package joblevel;

import java.util.ArrayList;

public class Controller {

    public static Controller controller;

    private ArrayList<Model> models;

    public static void Controller() {
        controller = new Controller();
        controller.models = new ArrayList<Model>();
    }

    public void add(Model jobLevel) {
        models.add(jobLevel);
    }

    public Model getModel(String text) {
        for(Model model: models) {
            if (model.getName().equals(text)) {
                return model;
            }
        }
        return null;
    }

    public ArrayList<Model> getModels() {
        return models;
    }

    public void setModels(ArrayList<Model> models) {
        this.models = models;
    }
}
```



```
}  
}
```

Model.java

```
package joblevel;
```

```
import java.io.Serializable;
```

```
import java.util.Objects;
```

```
public class Model implements Serializable {
```

```
    private String name;
```

```
    public Model(String name) {
```

```
        this.name = name;
```

```
    }
```

```
    public String getName() {
```

```
        return name;
```

```
    }
```

```
    public void setName(String name) {
```

```
        this.name = name;
```

```
    }
```

```
    @Override
```

```
    public boolean equals(Object o) {
```

```
        if (this == o) return true;
```

```
        if (!(o instanceof Model)) return false;
```

```
        Model model = (Model) o;
```

```
        return Objects.equals(getName(), model.getName());
```

```
    }
```

```
    @Override
```

```

    public int hashCode() {
        return Objects.hash(getName());
    }

    @Override
    public String toString() {
        return name;
    }
}

```

Package [jobtype](#)

[Controller.java](#)

```

package jobtype;

import java.util.ArrayList;

public class Controller {

    public static Controller controller;

    private ArrayList<Model> models;

    public static void Controller() {
        controller = new Controller();
        controller.models = new ArrayList<Model>();
    }

    public void add(Model jobType) {
        models.add(jobType);
    }

    public Model getModel(String text) {
        for(Model model: models) {
            if (model.getName().equals(text)) {

```

```

        return model;
    }
}
return null;
}

public ArrayList<Model> getModels() {
    return models;
}

public void setModels(ArrayList<Model> models) {
    this.models = models;
}
}

```

[Model.java](#)

```

package jobtype;

import java.io.Serializable;
import java.util.Objects;

public class Model implements Serializable {

    private String name;

    public Model(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {

```

```

        this.name = name;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof Model)) return false;
        Model model = (Model) o;
        return Objects.equals(getName(), model.getName());
    }

    @Override
    public int hashCode() {
        return Objects.hash(getName());
    }

    @Override
    public String toString() {
        return name;
    }
}

```

Package [location](#)

[Controller.java](#)

```

package location;

import java.util.ArrayList;

public class Controller {

    public static Controller controller;

```

```

private ArrayList<Model> models;

public static void Controller() {
    controller = new Controller();
    controller.models = new ArrayList<Model>();
}

public void add(Model jobType) {
    models.add(jobType);
}

public Model getModel(String text) {
    for(Model model: models) {
        if (model.getName().equals(text)) {
            return model;
        }
    }
    return null;
}

public ArrayList<Model> getModels() {
    return models;
}

public void setModels(ArrayList<Model> models) {
    this.models = models;
}
}

```

[Model.java](#)

package location;

```

import java.io.Serializable;
import java.util.Objects;

```

```

public class Model implements Serializable {

    private String name;

    public Model(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof Model)) return false;
        Model model = (Model) o;
        return Objects.equals(getName(), model.getName());
    }

    @Override
    public int hashCode() {
        return Objects.hash(getName());
    }

    @Override
    public String toString() {
        return name;
    }
}

```

