JAM

JAM

Job Application Management – a full stack application intended to manage job applications from the perspective of an applicant.
Created using following tools and components:

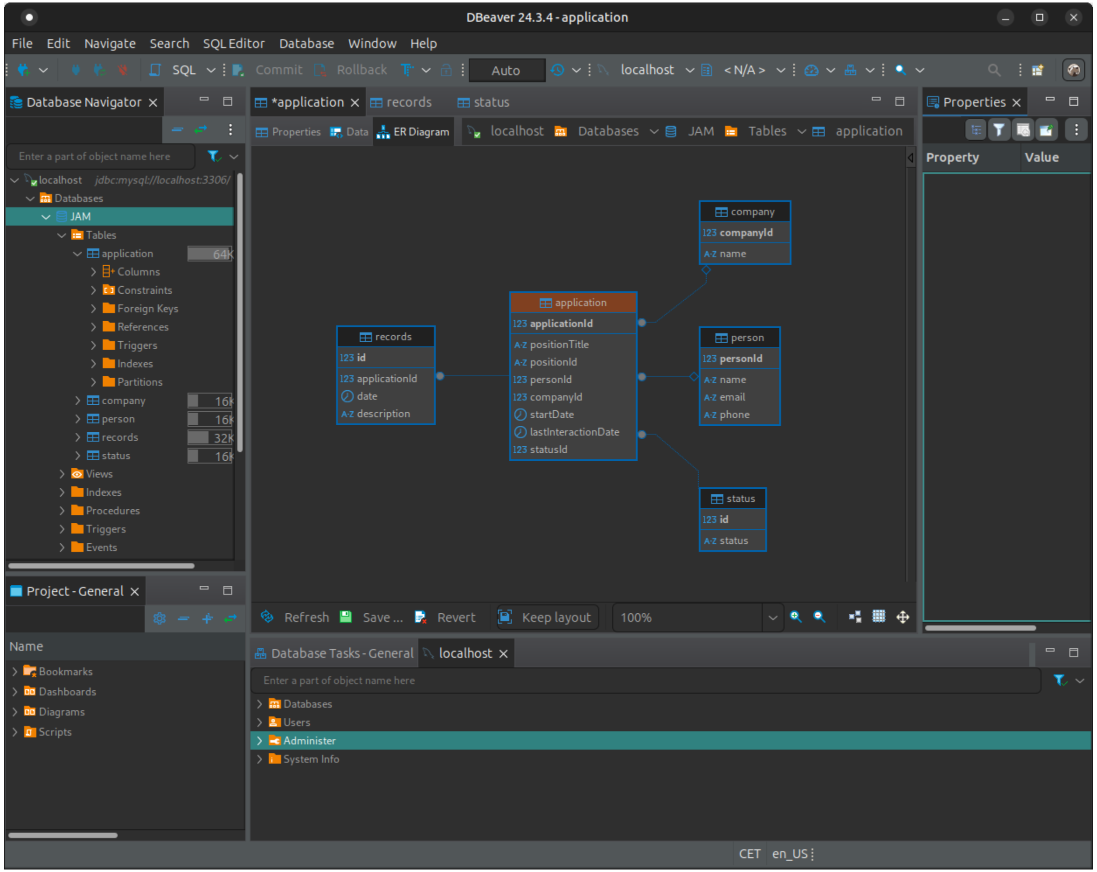
HIBERNATE IJ DBeaver POSTMAN MySQL™

Maven™ Java™ HTML CSS JS spring boot

Database layer

MySQL used as base layer database.

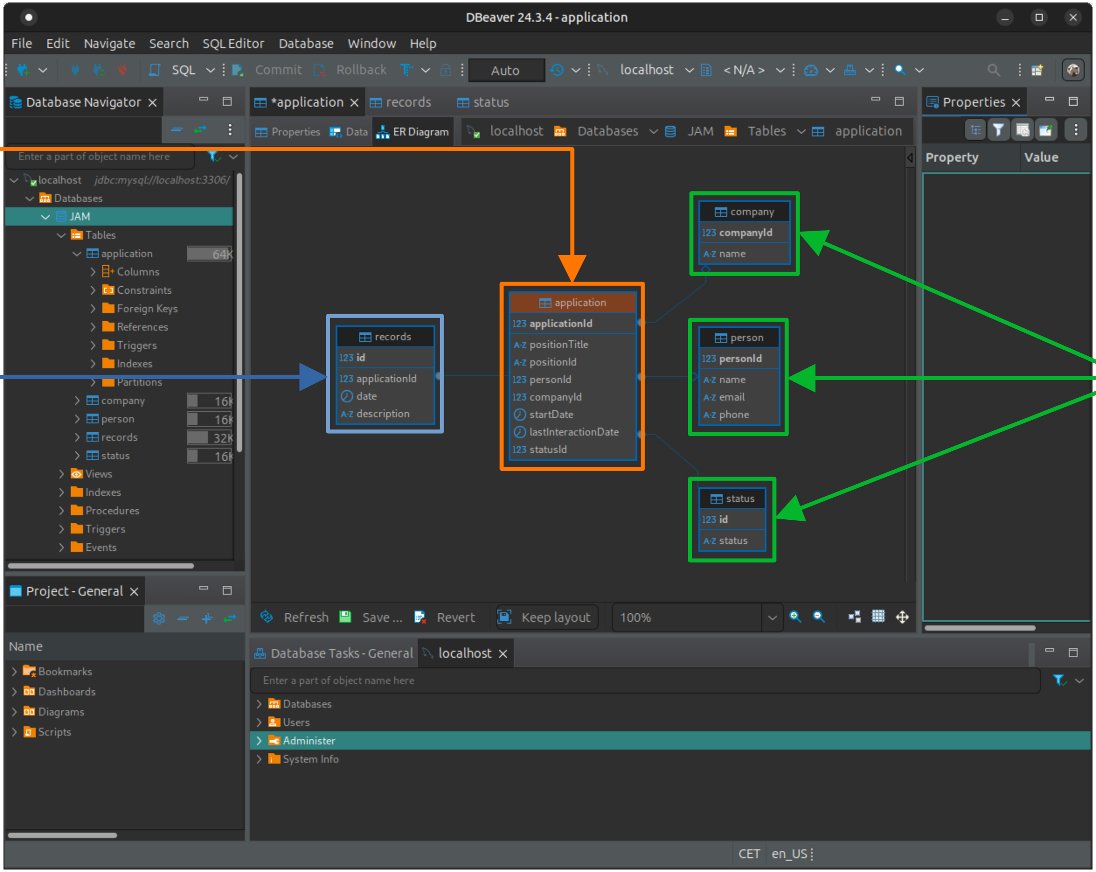
Layout of tables and views created using DBeaver



Database layer

“application“ table used to hold information about applications

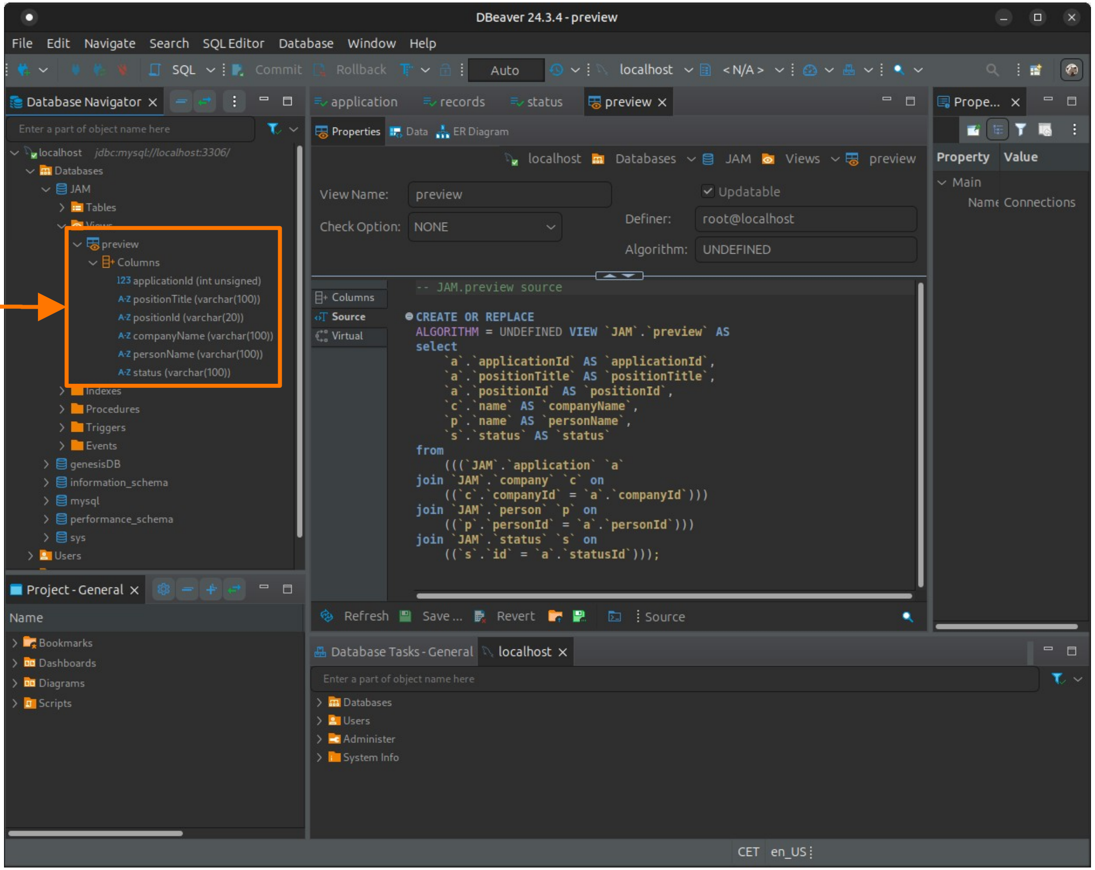
“records” table for various notes related to a particular application.



Tables holding information about persons, companies and status connected to “application” table via foreign keys

Database layer

“preview“ view created to ease access to data used for application card (see front end section)



```
localhost   jdbc:mysql://localhost:3306/
  Databases
    JAM
      Tables
      Views
        preview
          Columns
            applicationId (int unsigned)
            positionTitle (varchar(100))
            positionId (varchar(20))
            companyName (varchar(100))
            personName (varchar(100))
            status (varchar(100))
      Indexes
      Procedures
      Triggers
      Events
    genesisDB
    information_schema
    mysql
    performance_schema
    sys
  Users
```

View Name: preview

☑ Updatable

Check Option: NONE

Definer: root@localhost

Algorithm: UNDEFINED

```sql
-- JAM.preview source

CREATE OR REPLACE
ALGORITHM = UNDEFINED VIEW `JAM`.`preview` AS
select
    `a`.`applicationId` AS `applicationId`,
    `a`.`positionTitle` AS `positionTitle`,
    `a`.`positionId` AS `positionId`,
    `c`.`name` AS `companyName`,
    `p`.`name` AS `personName`,
    `s`.`status` AS `status`
from
    (((`JAM`.`application` `a`
join `JAM`.`company` `c` on
    ((`c`.`companyId` = `a`.`companyId`)))
join `JAM`.`person` `p` on
    ((`p`.`personId` = `a`.`personId`)))
join `JAM`.`status` `s` on
    ((`s`.`id` = `a`.`statusId`)));
```

MySQL

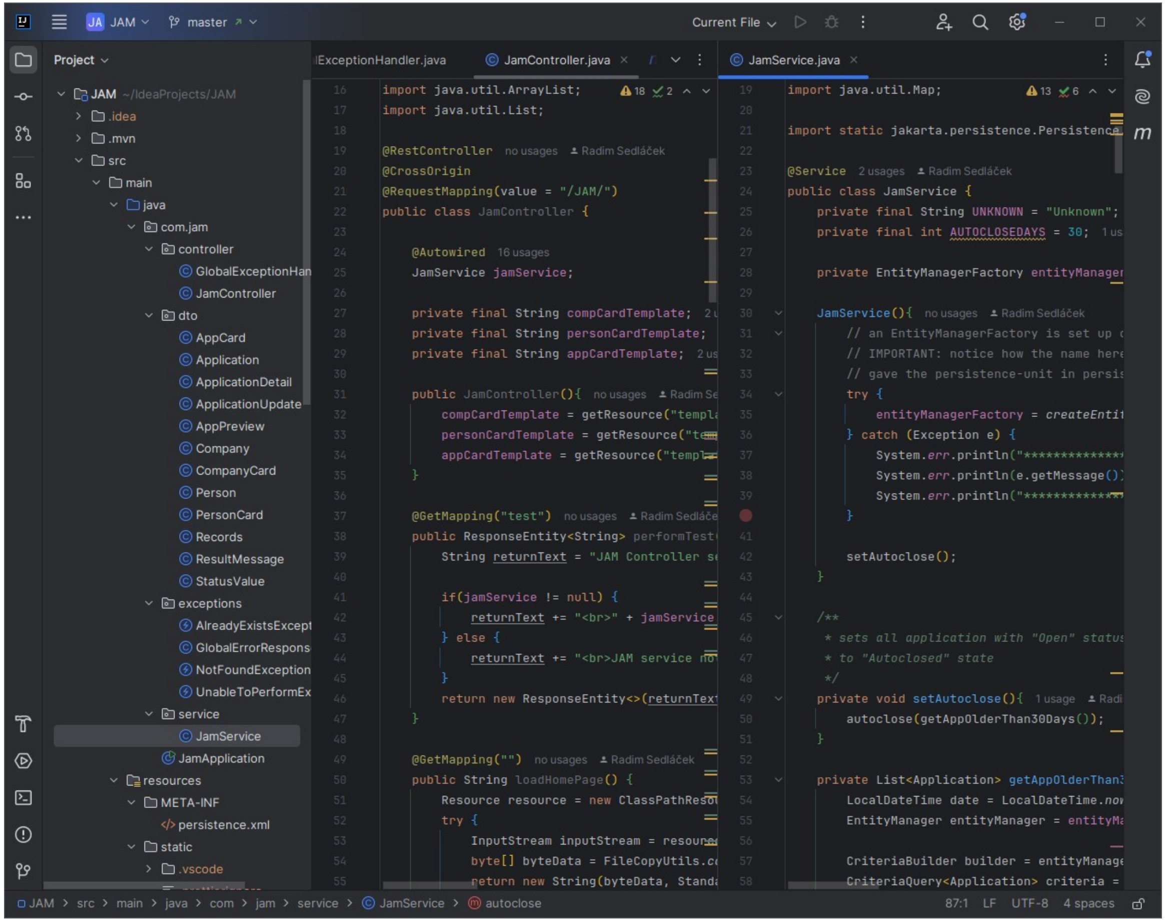
DBeaver

Back-end layer

- Java based
- Spring boot used for REST web service/web server creation
- Maven used for packaging
- Coded in Intellij Idea IDE (community edition)
- Hibernate used for interactions with database
- Postman used for testing

Back-end layer

Spring boot application/controller/service leveraged for requests processing



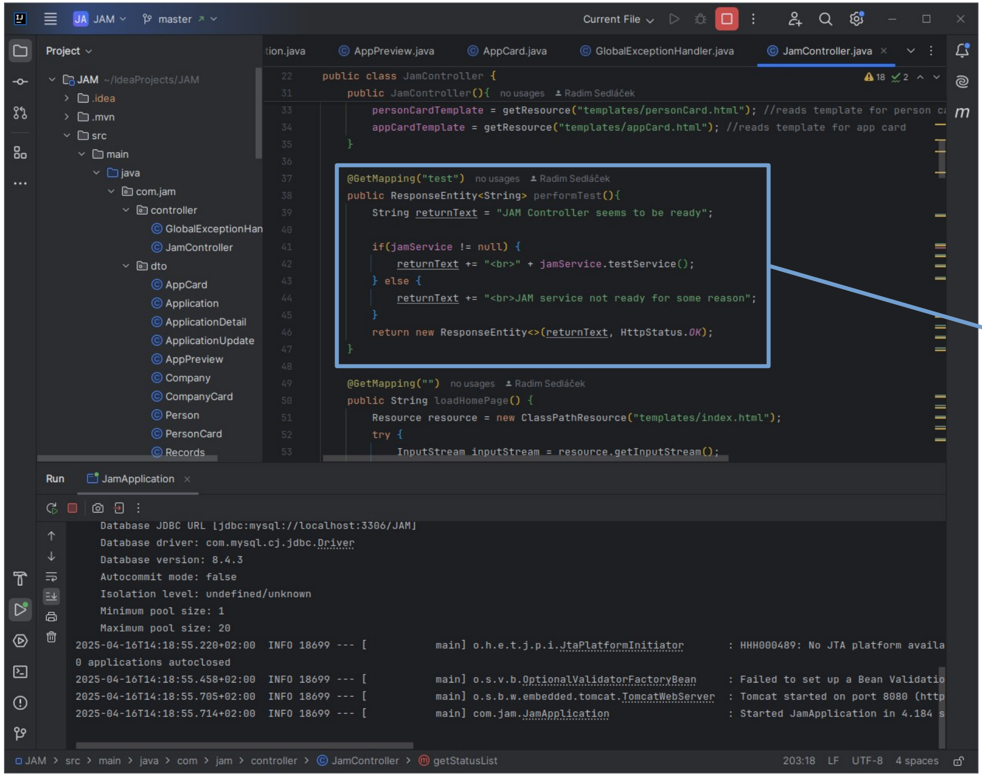
```
JAM ~/IdeaProjects/JAM
  .idea
  .mvn
  src
    main
      java
        com.jam
          controller
            GlobalExceptionHan
            JamController
          dto
            AppCard
            Application
            ApplicationDetail
            ApplicationUpdate
            AppPreview
            Company
            CompanyCard
            Person
            PersonCard
            Records
            ResultMessage
            StatusValue
          exceptions
            AlreadyExistsExcept
            GlobalErrorRespons
            NotFoundException
            UnableToPerformEx
          service
            JamService
          JamApplication
      resources
        META-INF
          persistence.xml
        static
          .vscode
```

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

@RestController  no usages  Radim Sedláček
@CrossOrigin
@RequestMapping(value = "/JAM/")
public class JamController {

    @Autowired  16 usages
    JamService jamService;

    private final String compCardTemplate;  2 u
    private final String personCardTemplate;
    private final String appCardTemplate;  2 us

    public JamController(){  no usages  Radim Se
        compCardTemplate = getResource("templa
        personCardTemplate = getResource("te
        appCardTemplate = getResource("templ
    }

    @GetMapping("test")  no usages  Radim Sedláče
    public ResponseEntity<String> performTest(
        String returnText = "JAM Controller se

        if(jamService != null) {
            returnText += "<br>" + jamService
        } else {
            returnText += "<br>JAM service no
        }
        return new ResponseEntity<>(returnText
    }

    @GetMapping("")  no usages  Radim Sedláček
    public String loadHomePage() {
        Resource resource = new ClassPathResou
        try {
            InputStream inputStream = resource
            byte[] byteData = FileCopyUtils.co
            return new String(byteData, Standa
```

```java
import java.util.Map;

import static jakarta.persistence.Persistence.

@Service  2 usages  Radim Sedláček
public class JamService {
    private final String UNKNOWN = "Unknown";
    private final int AUTOCLOSEDAYS = 30;  1 us

    private EntityManagerFactory entityManager

    JamService(){  no usages  Radim Sedláček
        // an EntityManagerFactory is set up c
        // IMPORTANT: notice how the name here
        // gave the persistence-unit in persis
        try {
            entityManagerFactory = createEntit
        } catch (Exception e) {
            System.err.println("**************
            System.err.println(e.getMessage()
            System.err.println("**************
        }

        setAutoclose();
    }

    /**
     * sets all application with "Open" status
     * to "Autoclosed" state
     */
    private void setAutoclose(){  1 usage  Radi
        autoclose(getAppOlderThan30Days());
    }

    private List<Application> getAppOlderThan3
        LocalDateTime date = LocalDateTime.now
        EntityManager entityManager = entityMa

        CriteriaBuilder builder = entityManage
        CriteriaQuery<Application> criteria =
```

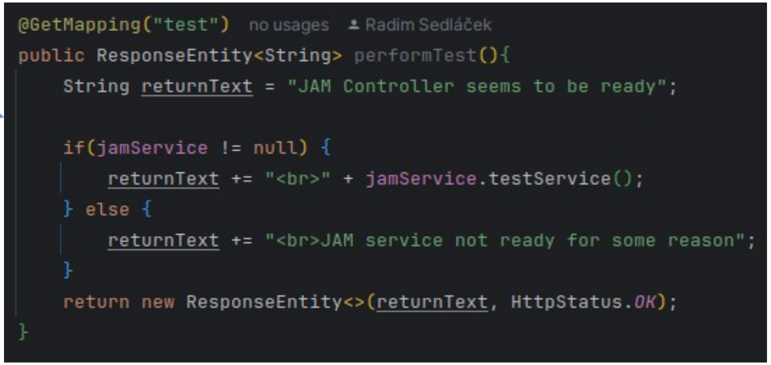
Back-end layer

A simple test of service and database readiness available on address “test”



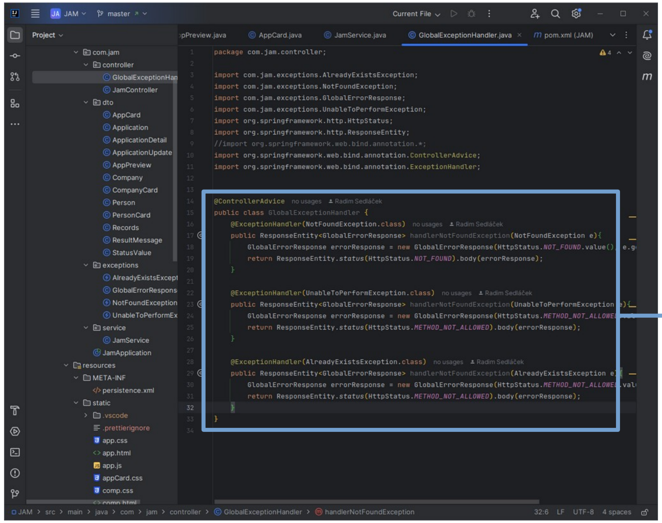
```java
public class JamController {
    public JamController(){  no usages  Radim Sedláček
        personCardTemplate = getResource("templates/personCard.html"); //reads template for person ca
        appCardTemplate = getResource("templates/appCard.html"); //reads template for app card
    }

    @GetMapping("test")  no usages  Radim Sedláček
    public ResponseEntity<String> performTest(){
        String returnText = "JAM Controller seems to be ready";

        if(jamService != null) {
            returnText += "<br>" + jamService.testService();
        } else {
            returnText += "<br>JAM service not ready for some reason";
        }
        return new ResponseEntity<>(returnText, HttpStatus.OK);
    }

    @GetMapping("")  no usages  Radim Sedláček
    public String loadHomePage() {
        Resource resource = new ClassPathResource("templates/index.html");
        try {
            InputStream inputStream = resource.getInputStream();
```

```
    Database JDBC URL [jdbc:mysql://localhost:3306/JAM]
    Database driver: com.mysql.cj.jdbc.Driver
    Database version: 8.4.3
    Autocommit mode: false
    Isolation level: undefined/unknown
    Minimum pool size: 1
    Maximum pool size: 20
2025-04-16T14:18:55.220+02:00  INFO 18699 --- [           main] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000489: No JTA platform availa
0 applications autoclosed
2025-04-16T14:18:55.458+02:00  INFO 18699 --- [           main] o.s.v.b.OptionalValidatorFactoryBean     : Failed to set up a Bean Validatio
2025-04-16T14:18:55.705+02:00  INFO 18699 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port 8080 (http
2025-04-16T14:18:55.714+02:00  INFO 18699 --- [           main] com.jam.JamApplication                   : Started JamApplication in 4.184 s
```

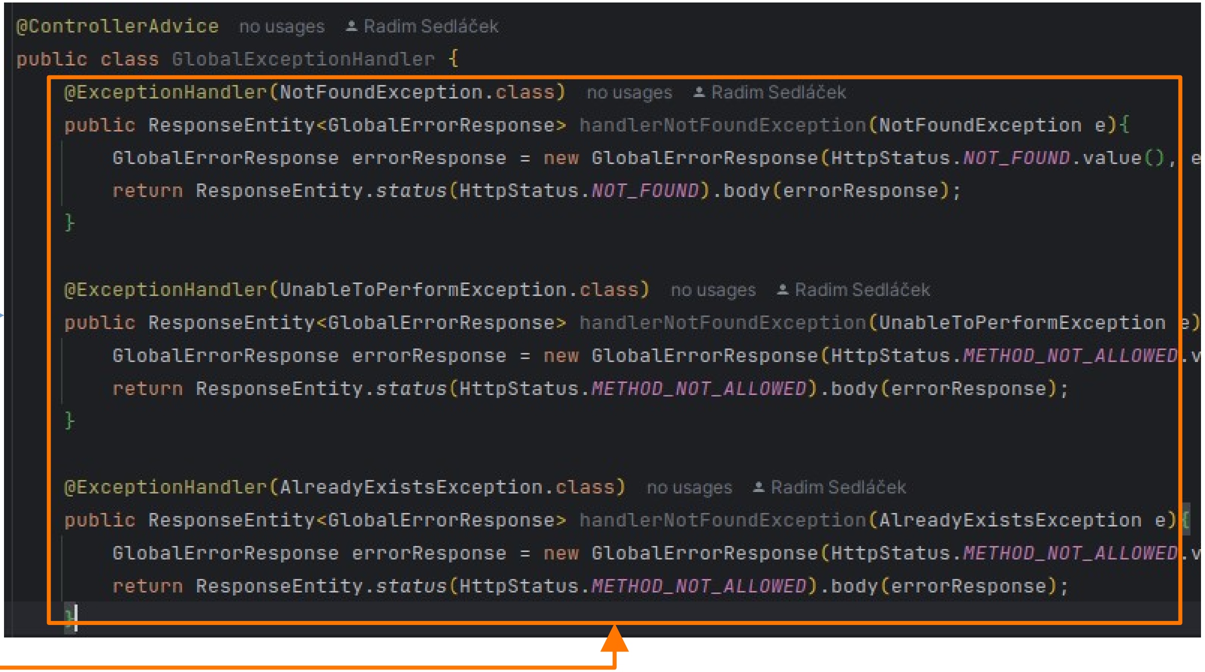


```java
@GetMapping("test")  no usages  Radim Sedláček
public ResponseEntity<String> performTest(){
    String returnText = "JAM Controller seems to be ready";

    if(jamService != null) {
        returnText += "<br>" + jamService.testService();
    } else {
        returnText += "<br>JAM service not ready for some reason";
    }
    return new ResponseEntity<>(returnText, HttpStatus.OK);
}
```

Back-end layer



```java
package com.jam.controller;

import com.jam.exceptions.AlreadyExistsException;
import com.jam.exceptions.NotFoundException;
import com.jam.exceptions.GlobalErrorResponse;
import com.jam.exceptions.UnableToPerformException;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
//import org.springframework.web.bind.annotation.*;
import org.springframework.web.bind.annotation.ControllerAdvice;
import org.springframework.web.bind.annotation.ExceptionHandler;


@ControllerAdvice  no usages  Radim Sedláček
public class GlobalExceptionHandler {
    @ExceptionHandler(NotFoundException.class)  no usages  Radim Sedláček
    public ResponseEntity<GlobalErrorResponse> handlerNotFoundException(NotFoundException e){
        GlobalErrorResponse errorResponse = new GlobalErrorResponse(HttpStatus.NOT_FOUND.value(), e.g
        return ResponseEntity.status(HttpStatus.NOT_FOUND).body(errorResponse);
    }

    @ExceptionHandler(UnableToPerformException.class)  no usages  Radim Sedláček
    public ResponseEntity<GlobalErrorResponse> handlerNotFoundException(UnableToPerformException e){
        GlobalErrorResponse errorResponse = new GlobalErrorResponse(HttpStatus.METHOD_NOT_ALLOWE
        return ResponseEntity.status(HttpStatus.METHOD_NOT_ALLOWED).body(errorResponse);
    }

    @ExceptionHandler(AlreadyExistsException.class)  no usages  Radim Sedláček
    public ResponseEntity<GlobalErrorResponse> handlerNotFoundException(AlreadyExistsException e){
        GlobalErrorResponse errorResponse = new GlobalErrorResponse(HttpStatus.METHOD_NOT_ALLOWED.val
        return ResponseEntity.status(HttpStatus.METHOD_NOT_ALLOWED).body(errorResponse);
    }
}
```



```java
@ControllerAdvice  no usages  Radim Sedláček
public class GlobalExceptionHandler {
    @ExceptionHandler(NotFoundException.class)  no usages  Radim Sedláček
    public ResponseEntity<GlobalErrorResponse> handlerNotFoundException(NotFoundException e){
        GlobalErrorResponse errorResponse = new GlobalErrorResponse(HttpStatus.NOT_FOUND.value(), e
        return ResponseEntity.status(HttpStatus.NOT_FOUND).body(errorResponse);
    }

    @ExceptionHandler(UnableToPerformException.class)  no usages  Radim Sedláček
    public ResponseEntity<GlobalErrorResponse> handlerNotFoundException(UnableToPerformException e)
        GlobalErrorResponse errorResponse = new GlobalErrorResponse(HttpStatus.METHOD_NOT_ALLOWED.v
        return ResponseEntity.status(HttpStatus.METHOD_NOT_ALLOWED).body(errorResponse);
    }

    @ExceptionHandler(AlreadyExistsException.class)  no usages  Radim Sedláček
    public ResponseEntity<GlobalErrorResponse> handlerNotFoundException(AlreadyExistsException e){
        GlobalErrorResponse errorResponse = new GlobalErrorResponse(HttpStatus.METHOD_NOT_ALLOWED.v
        return ResponseEntity.status(HttpStatus.METHOD_NOT_ALLOWED).body(errorResponse);
    }
```

Spring boot @ExceptionHandler leveraged for error processing

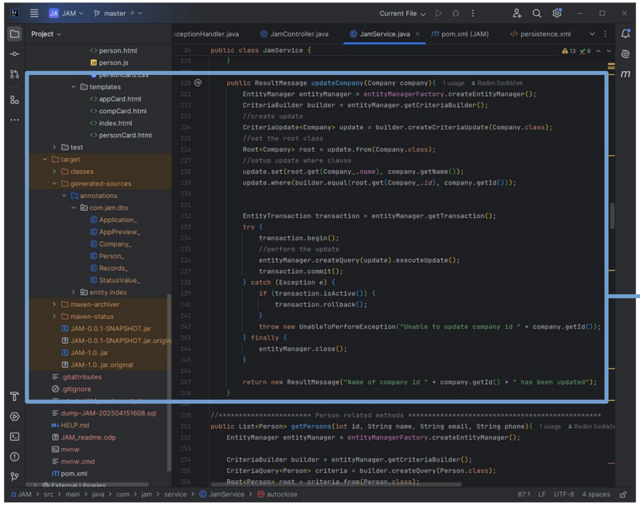
HIBERNATE

spring boot

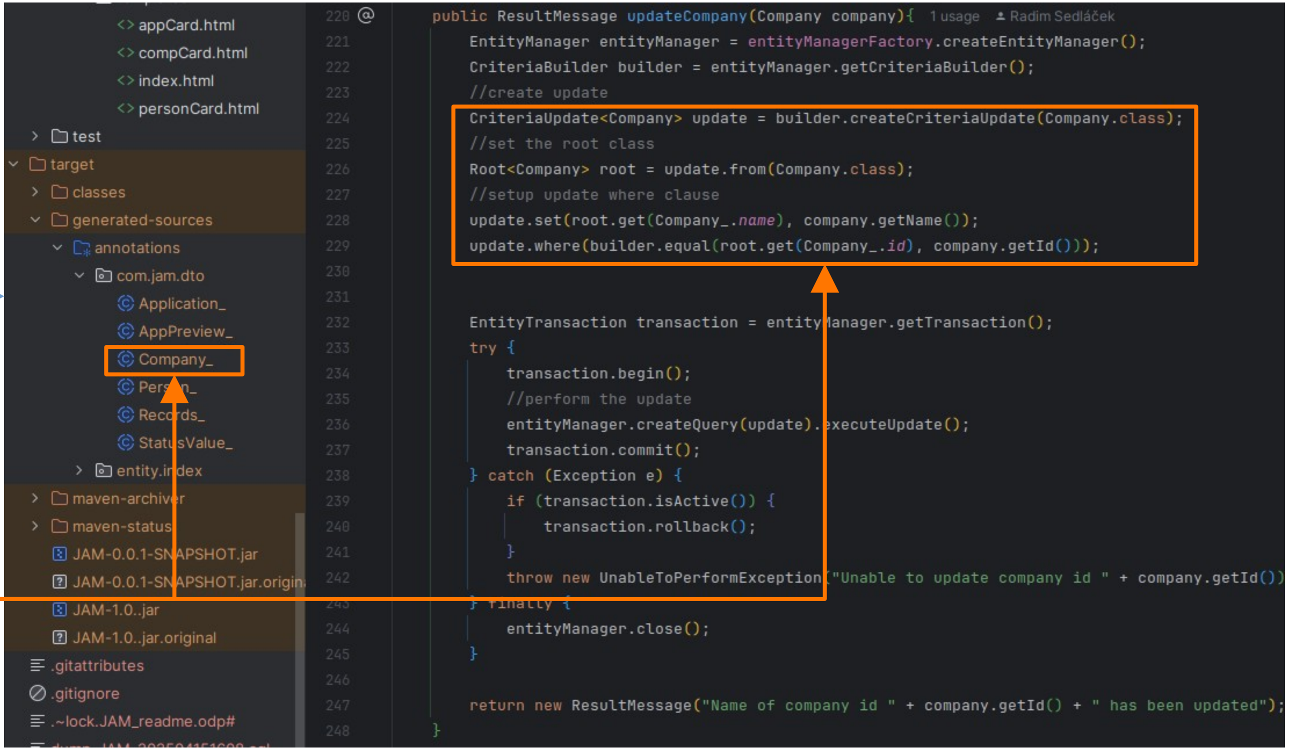
Java

IJ

Back-end layer



```java
public class JamService {
    }

    public ResultMessage updateCompany(Company company){  1 usage  Radim Sedláček
        EntityManager entityManager = entityManagerFactory.createEntityManager();
        CriteriaBuilder builder = entityManager.getCriteriaBuilder();
        //create update
        CriteriaUpdate<Company> update = builder.createCriteriaUpdate(Company.class);
        //set the root class
        Root<Company> root = update.from(Company.class);
        //setup update where clause
        update.set(root.get(Company_.name), company.getName());
        update.where(builder.equal(root.get(Company_.id), company.getId()));


        EntityTransaction transaction = entityManager.getTransaction();
        try {
            transaction.begin();
            //perform the update
            entityManager.createQuery(update).executeUpdate();
            transaction.commit();
        } catch (Exception e) {
            if (transaction.isActive()) {
                transaction.rollback();
            }
            throw new UnableToPerformException("Unable to update company id " + company.getId());
        } finally {
            entityManager.close();
        }

        return new ResultMessage("Name of company id " + company.getId() + " has been updated");
    }

//********************** Person related methods ******************************************
public List<Person> getPersons(int id, String name, String email, String phone){  1 usage  Radim Sedláče
    EntityManager entityManager = entityManagerFactory.createEntityManager();

    CriteriaBuilder builder = entityManager.getCriteriaBuilder();
    CriteriaQuery<Person> criteria = builder.createQuery(Person.class);
    Root<Person> root = criteria.from(Person.class);
```

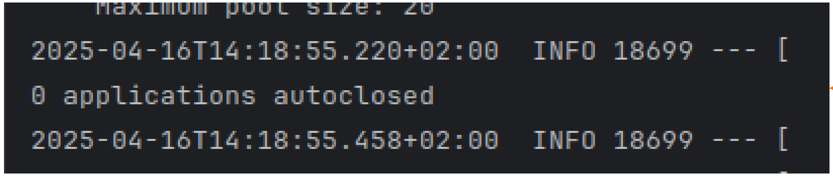


```java
public ResultMessage updateCompany(Company company){  1 usage  Radim Sedláček
    EntityManager entityManager = entityManagerFactory.createEntityManager();
    CriteriaBuilder builder = entityManager.getCriteriaBuilder();
    //create update
    CriteriaUpdate<Company> update = builder.createCriteriaUpdate(Company.class);
    //set the root class
    Root<Company> root = update.from(Company.class);
    //setup update where clause
    update.set(root.get(Company_.name), company.getName());
    update.where(builder.equal(root.get(Company_.id), company.getId()));


    EntityTransaction transaction = entityManager.getTransaction();
    try {
        transaction.begin();
        //perform the update
        entityManager.createQuery(update).executeUpdate();
        transaction.commit();
    } catch (Exception e) {
        if (transaction.isActive()) {
            transaction.rollback();
        }
        throw new UnableToPerformException("Unable to update company id " + company.getId())
    } finally {
        entityManager.close();
    }

    return new ResultMessage("Name of company id " + company.getId() + " has been updated");
}
```

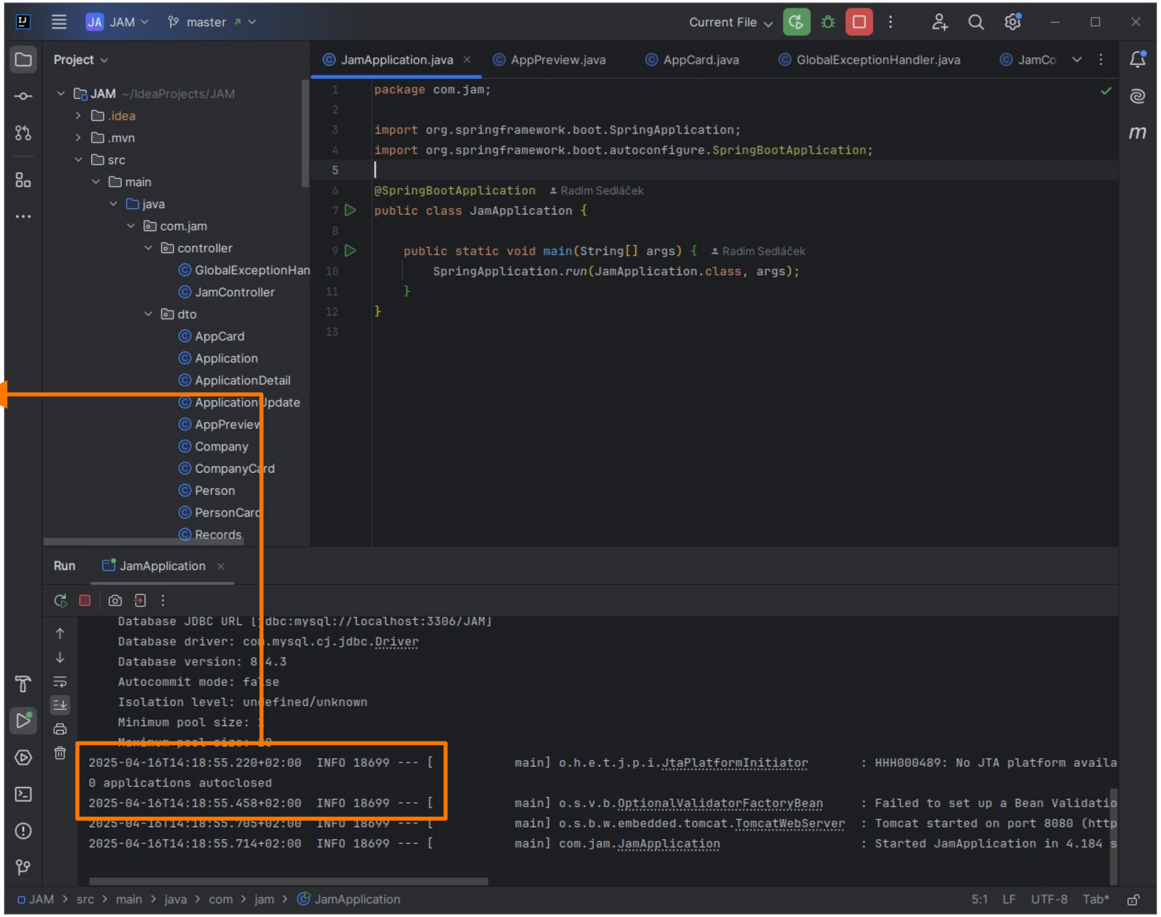
Hibernate static meta-model leveraged to enforce type safety

Back-end layer

During every start the service checks database for application with idle time longer than 30 days and auto closes them



```
    Maximum pool size: 20
2025-04-16T14:18:55.220+02:00  INFO 18699 --- [
0 applications autoclosed
2025-04-16T14:18:55.458+02:00  INFO 18699 --- [
```

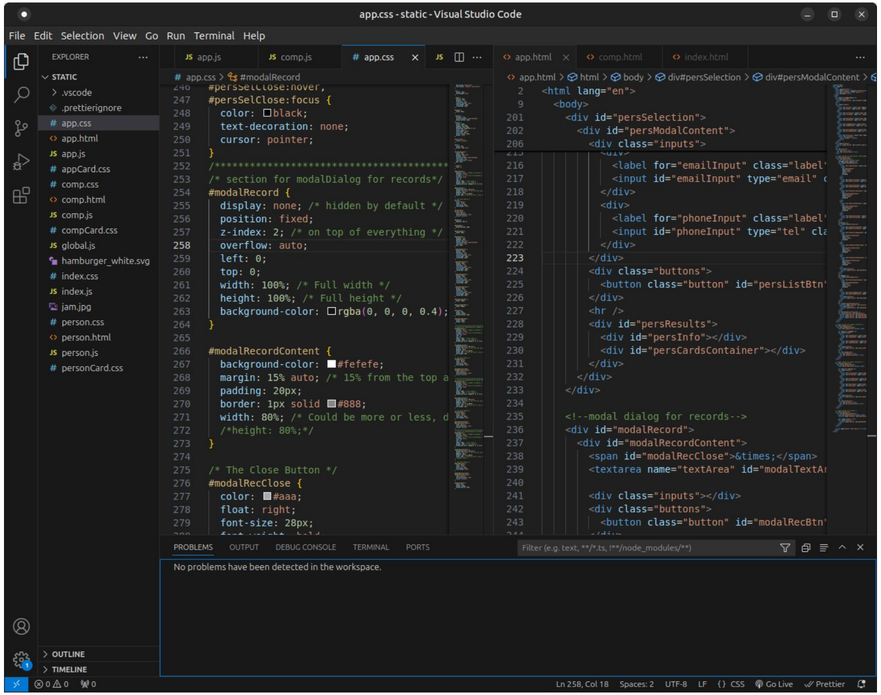


```java
package com.jam;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication  Radim Sedláček
public class JamApplication {

    public static void main(String[] args) {  Radim Sedláček
        SpringApplication.run(JamApplication.class, args);
    }
}
```

```
    Database JDBC URL [jdbc:mysql://localhost:3306/JAM]
    Database driver: com.mysql.cj.jdbc.Driver
    Database version: 8.4.3
    Autocommit mode: false
    Isolation level: undefined/unknown
    Minimum pool size: 1
    Maximum pool size: 20
2025-04-16T14:18:55.220+02:00  INFO 18699 --- [           main] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000489: No JTA platform availa
0 applications autoclosed
2025-04-16T14:18:55.458+02:00  INFO 18699 --- [           main] o.s.v.b.OptionalValidatorFactoryBean     : Failed to set up a Bean Validatio
2025-04-16T14:18:55.705+02:00  INFO 18699 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port 8080 (http
2025-04-16T14:18:55.714+02:00  INFO 18699 --- [           main] com.jam.JamApplication                   : Started JamApplication in 4.184 s
```

Front-end layer

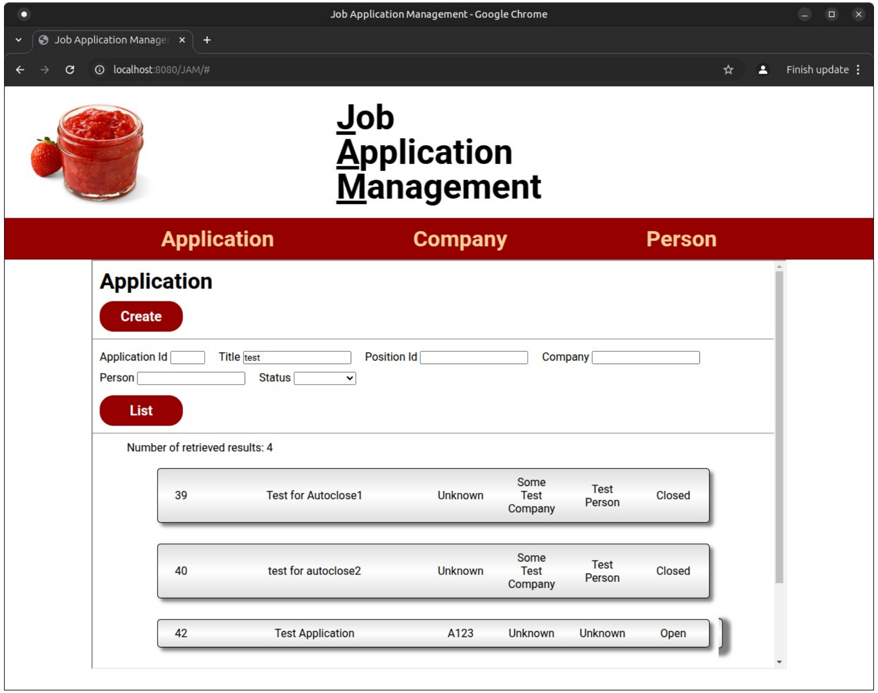
- HTML/CSS/Javascript based web page
- Coded in Visual Studio Code



```css
#persSelClose:hover,
#persSelClose:focus {
  color: black;
  text-decoration: none;
  cursor: pointer;
}
/****************************************
/* section for modalDialog for records*/
#modalRecord {
  display: none; /* hidden by default */
  position: fixed;
  z-index: 2; /* on top of everything */
  overflow: auto;
  left: 0;
  top: 0;
  width: 100%; /* Full width */
  height: 100%; /* Full height */
  background-color: rgba(0, 0, 0, 0.4);
}

#modalRecordContent {
  background-color: #fefefe;
  margin: 15% auto; /* 15% from the top a
  padding: 20px;
  border: 1px solid #888;
  width: 80%; /* Could be more or less, d
  /*height: 80%;*/
}

/* The Close Button */
#modalRecClose {
  color: #aaa;
  float: right;
  font-size: 28px;
```

```html
<html lang="en">
  <body>
    <div id="persSelection">
      <div id="persModalContent">
        <div class="inputs">
            <label for="emailInput" class="label'
            <input id="emailInput" type="email" c
          </div>
          <div>
            <label for="phoneInput" class="label'
            <input id="phoneInput" type="tel" cla
          </div>
        </div>
        <div class="buttons">
          <button class="button" id="persListBtn'
        </div>
        <hr />
        <div id="persResults">
          <div id="persInfo"></div>
          <div id="persCardsContainer"></div>
        </div>
      </div>
    </div>

    <!--modal dialog for records-->
    <div id="modalRecord">
      <div id="modalRecordContent">
        <span id="modalRecClose">&times;</span>
        <textarea name="textArea" id="modalTextAr

        <div class="inputs"></div>
        <div class="buttons">
          <button class="button" id="modalRecBtn'
```

No problems have been detected in the workspace.



Job Application Management

Application | Company | Person

Application

Create

Application Id
Title test
Position Id
Company
Person
Status

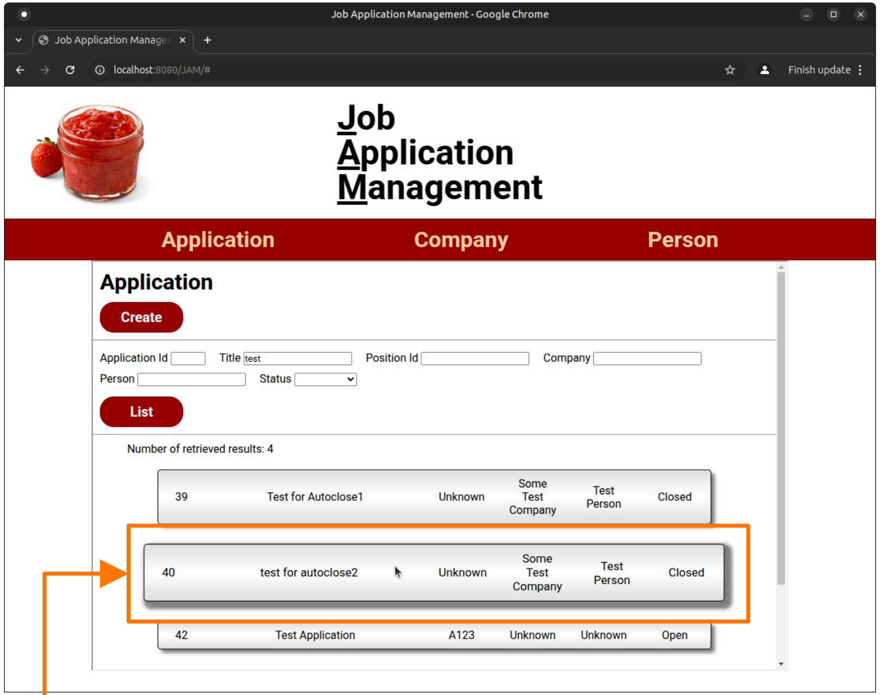
List

Number of retrieved results: 4

39	Test for Autoclose1	Unknown	Some Test Company	Test Person	Closed
40	test for autoclose2	Unknown	Some Test Company	Test Person	Closed
42	Test Application	A123	Unknown	Unknown	Open

Front-end layer

Responsive design for use on desktop, tablet as well as smart phone

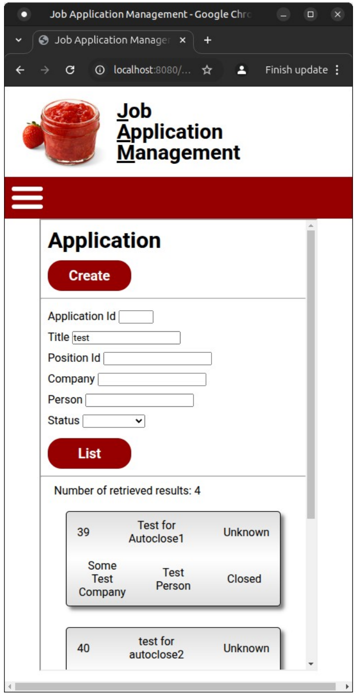


Application

Application Id ▭ Title test Position Id ▭ Company ▭
Person ▭ Status ▭

Number of retrieved results: 4

39	Test for Autoclose1	Unknown	Some Test Company	Test Person	Closed
40	test for autoclose2	Unknown	Some Test Company	Test Person	Closed
42	Test Application	A123	Unknown	Unknown	Open


Job Application Management - Google Chr
Job Application Manage
localhost:8080/...
Finish update
Job Application Management
Create
List

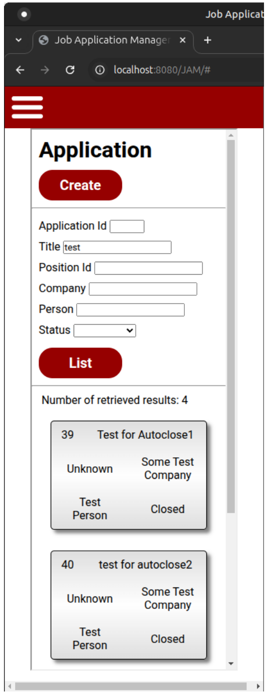
Application

Application Id ▭
Title test
Position Id ▭
Company ▭
Person ▭
Status ▭

Number of retrieved results: 4

39 Test for Autoclose1 Unknown
Some Test Company Test Person Closed

40 test for autoclose2 Unknown



Application

Application Id ▭
Title test
Position Id ▭
Company ▭
Person ▭
Status ▭

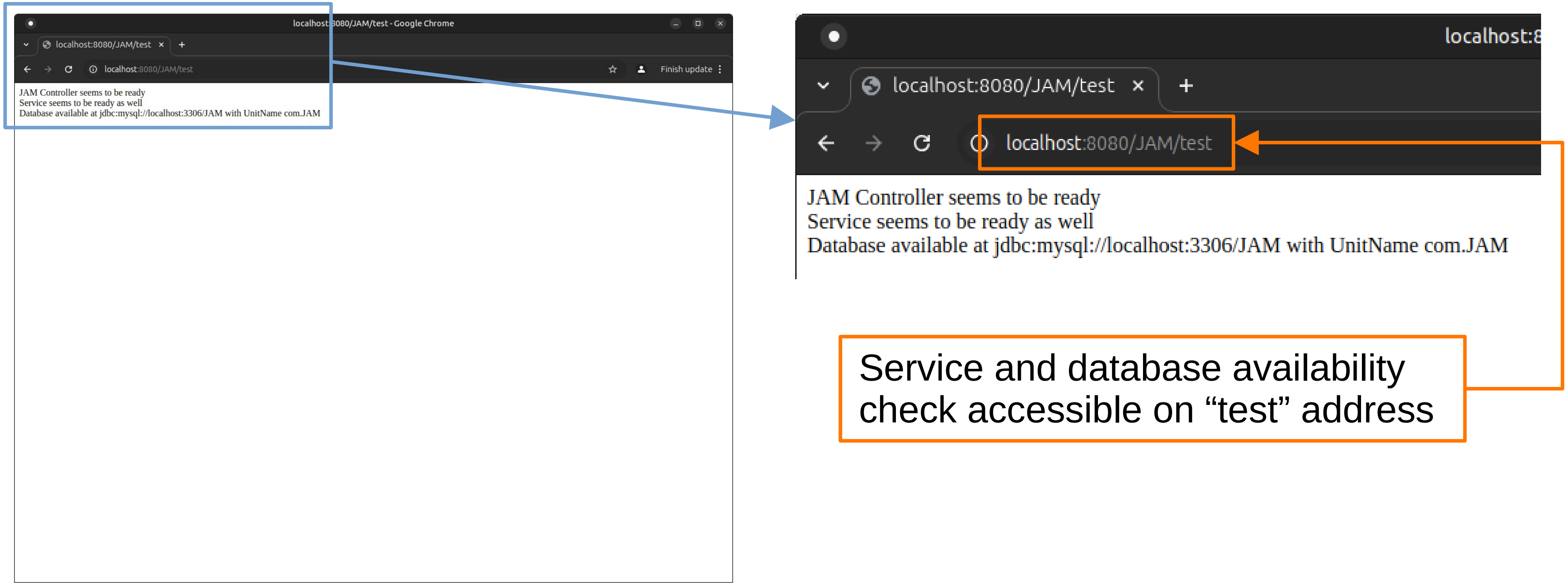
Number of retrieved results: 4

39 Test for Autoclose1
Unknown Some Test Company
Test Person Closed

40 test for autoclose2
Unknown Some Test Company
Test Person Closed

3D like effect when hovering over a card in the result list

Front-end layer

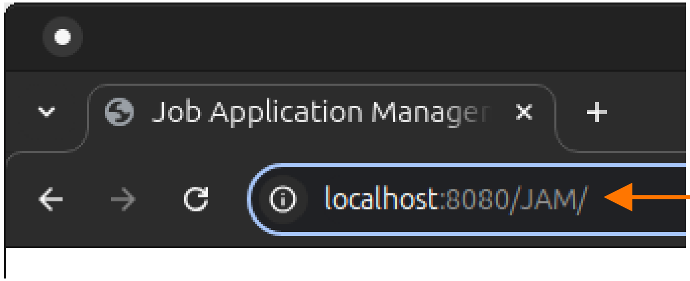


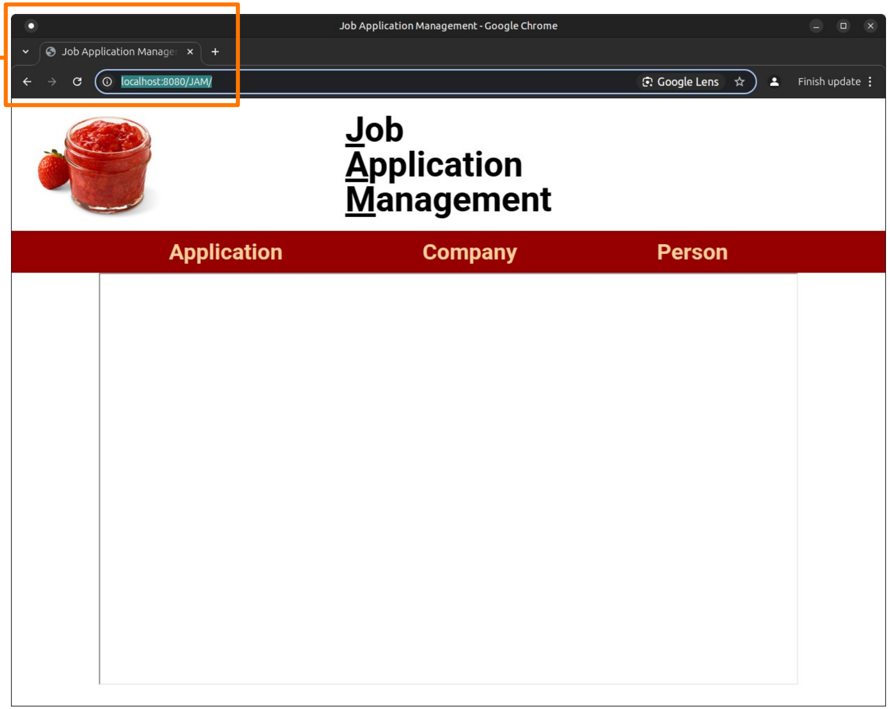
```
JAM Controller seems to be ready
Service seems to be ready as well
Database available at jdbc:mysql://localhost:3306/JAM with UnitName com.JAM
```

```
JAM Controller seems to be ready
Service seems to be ready as well
Database available at jdbc:mysql://localhost:3306/JAM with UnitName com.JAM
```

Service and database availability check accessible on “test” address

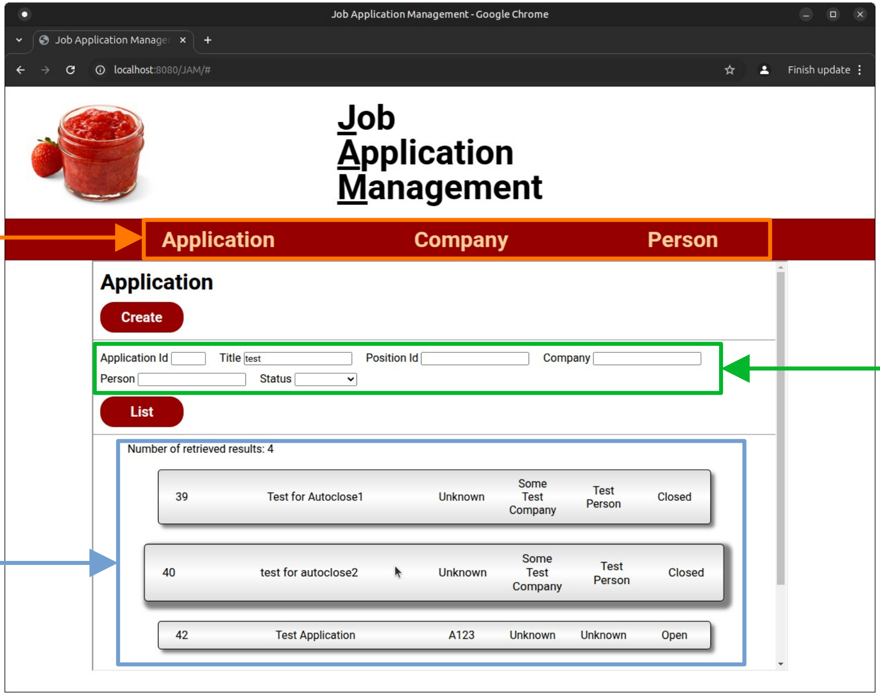
Front-end layer





When the server is running, the main web page itself available on “JAM” address

Front-end layer



Application

Application Id Title test Position Id Company
Person Status

Number of retrieved results: 4

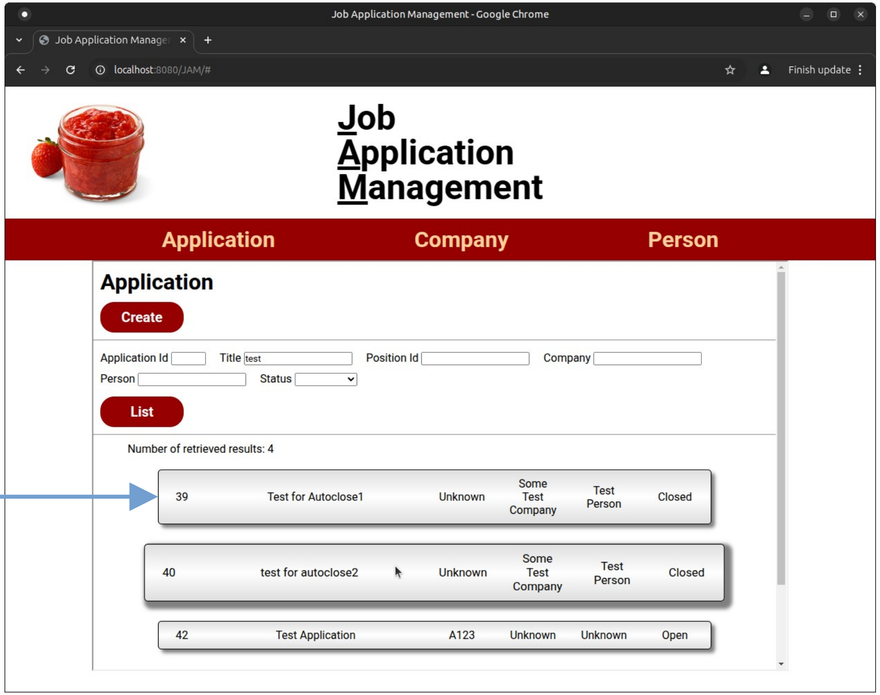
39	Test for Autoclose1	Unknown	Some Test Company	Test Person	Closed
40	test for autoclose2	Unknown	Some Test Company	Test Person	Closed
42	Test Application	A123	Unknown	Unknown	Open

Area with entity to create/edit/delete selection

Area with filters available for each entity

Result list area

Front-end layer



Job Application Management

Application Company Person

Application

Create

Application Id
Title test
Position Id
Company
Person
Status

List

Number of retrieved results: 4

39	Test for Autoclose1	Unknown	Some Test Company	Test Person	Closed
40	test for autoclose2	Unknown	Some Test Company	Test Person	Closed
42	Test Application	A123	Unknown	Unknown	Open

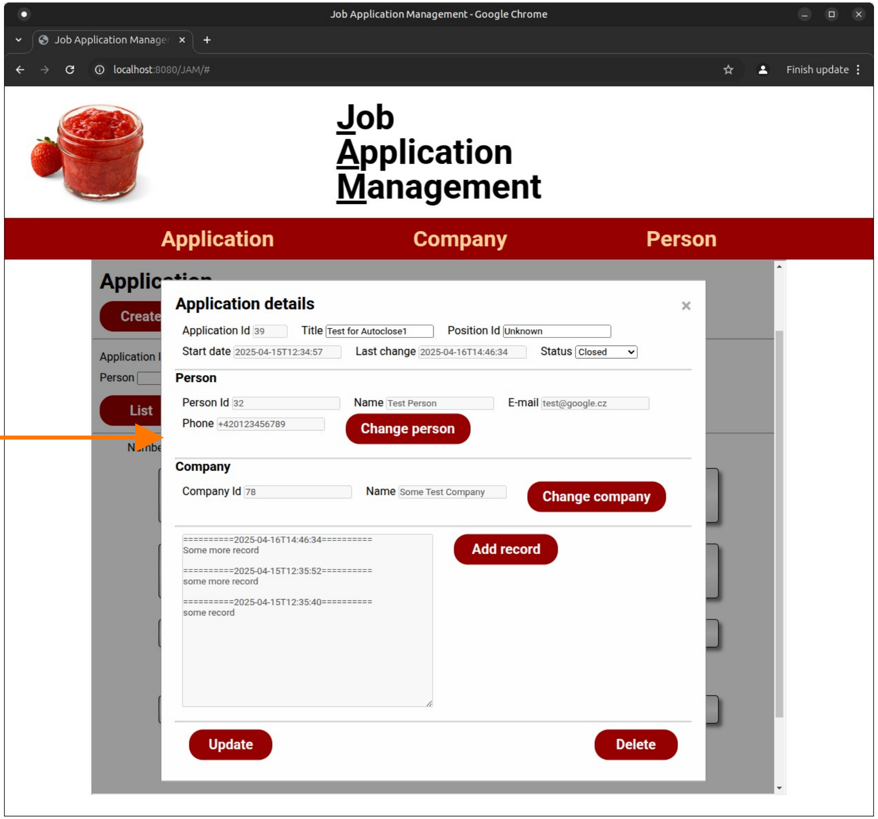
Data for each application card retrieved via “preview” view (see back-end layer section)

Clicking on a card will select given application for update

Front-end layer

After selecting an application, modal window like dialog appears to allow changes.

Available changes include change of contact person or company as well as recording notes, or deletion of the application



Application details

Application Id 39 Title Test for Autoclose1 Position Id Unknown
Start date 2025-04-15T12:34:57 Last change 2025-04-16T14:46:34 Status Closed

Person

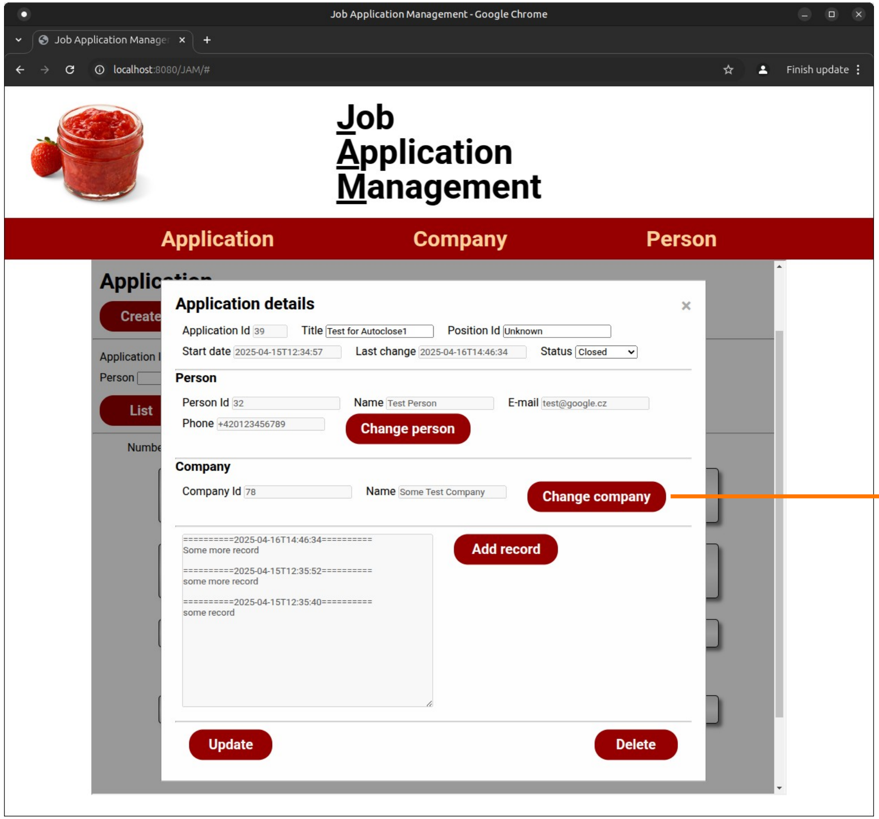
Person Id 32 Name Test Person E-mail test@google.cz
Phone +420123456789

Company

Company Id 78 Name Some Test Company

```
==========2025-04-16T14:46:34==========
Some more record

==========2025-04-15T12:35:52==========
some more record

==========2025-04-15T12:35:40==========
some record
```

Front-end layer



Job Application Management - Google Chrome

localhost:8080/JAM/#

Job Application Management

Application | Company | Person

Application details

Application Id 39 | Title Test for Autoclose1 | Position Id Unknown

Start date 2025-04-15T12:34:57 | Last change 2025-04-16T14:46:34 | Status Closed

Person

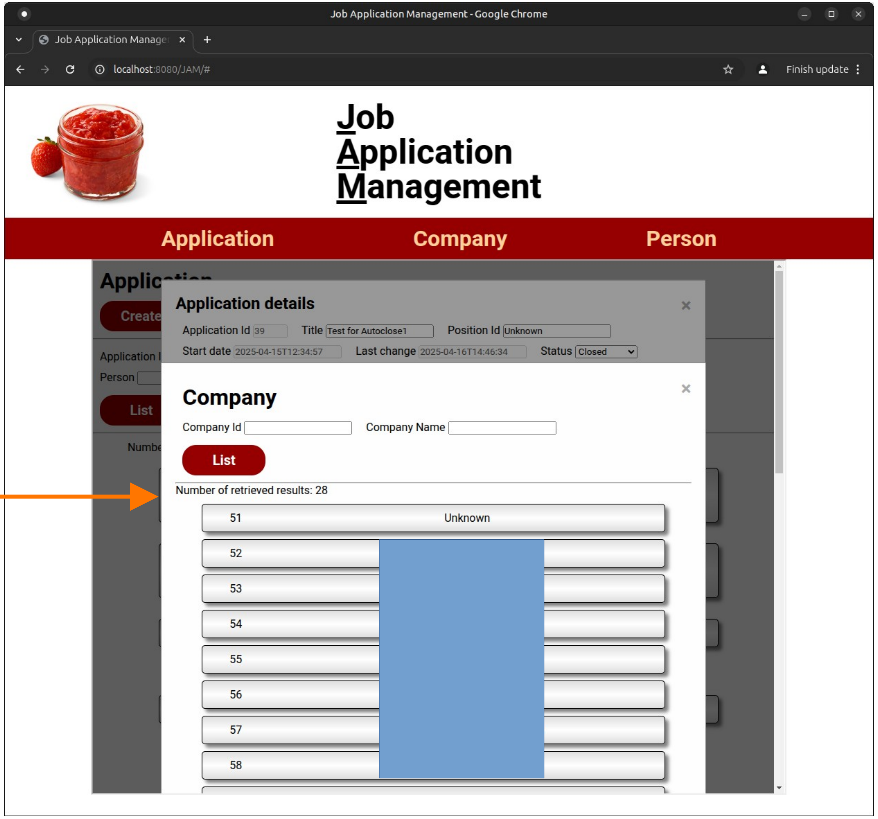
Person Id 32 | Name Test Person | E-mail test@google.cz

Phone +420123456789

Company

Company Id 78 | Name Some Test Company

```
==========2025-04-16T14:46:34==========
Some more record

==========2025-04-15T12:35:52==========
some more record

==========2025-04-15T12:35:40==========
some record
```



Job Application Management - Google Chrome

localhost:8080/JAM/#

Job Application Management

Application | Company | Person

Application details

Application Id 39 | Title Test for Autoclose1 | Position Id Unknown

Start date 2025-04-15T12:34:57 | Last change 2025-04-16T14:46:34 | Status Closed

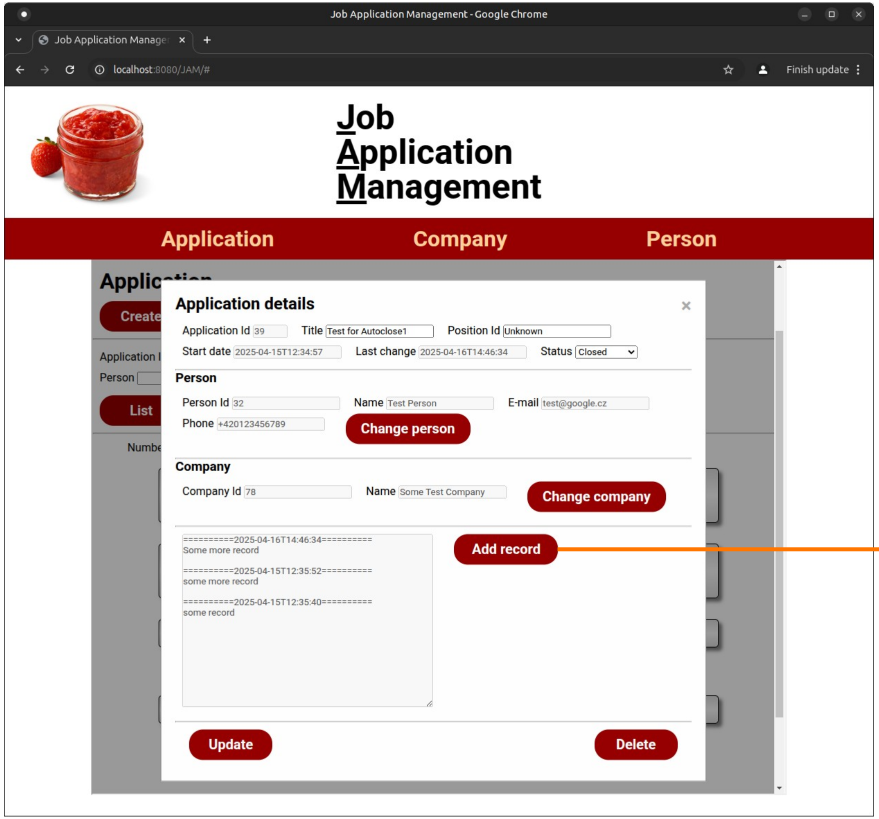
Company

Company Id | Company Name

Number of retrieved results: 28

51	Unknown
52	
53	
54	
55	
56	
57	
58	

Front-end layer



Application details

Application Id 39 Title Test for Autoclose1 Position Id Unknown

Start date 2025-04-15T12:34:57 Last change 2025-04-16T14:46:34 Status Closed

Person

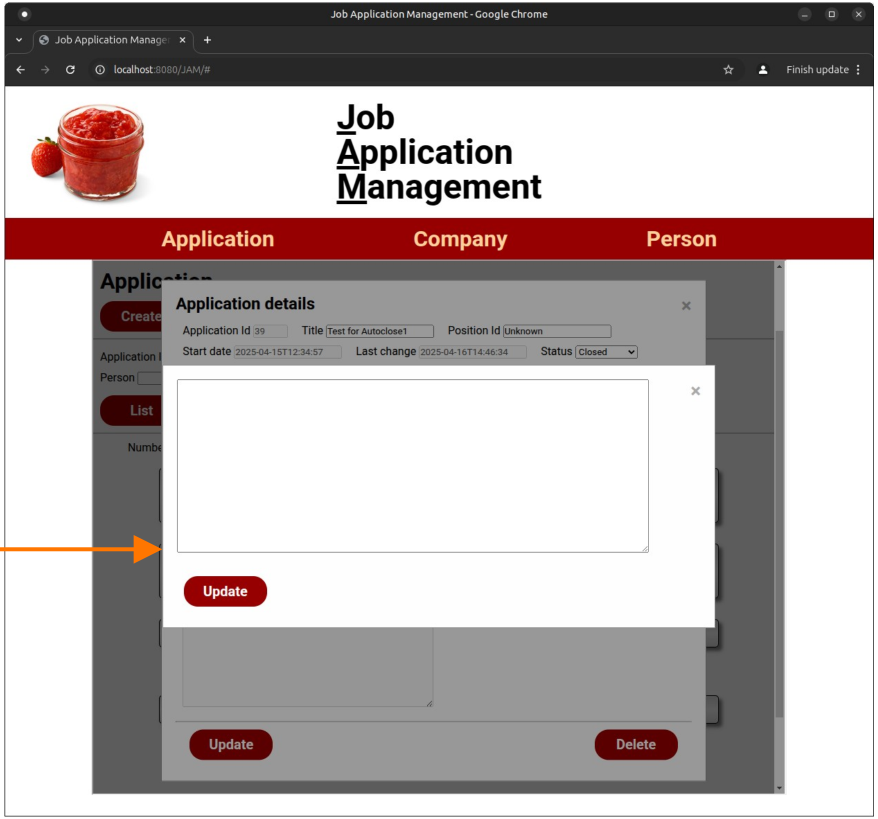
Person Id 32 Name Test Person E-mail test@google.cz

Phone +420123456789

Company

Company Id 78 Name Some Test Company

```
==========2025-04-16T14:46:34==========
Some more record

==========2025-04-15T12:35:52==========
some more record

==========2025-04-15T12:35:40==========
some record
```



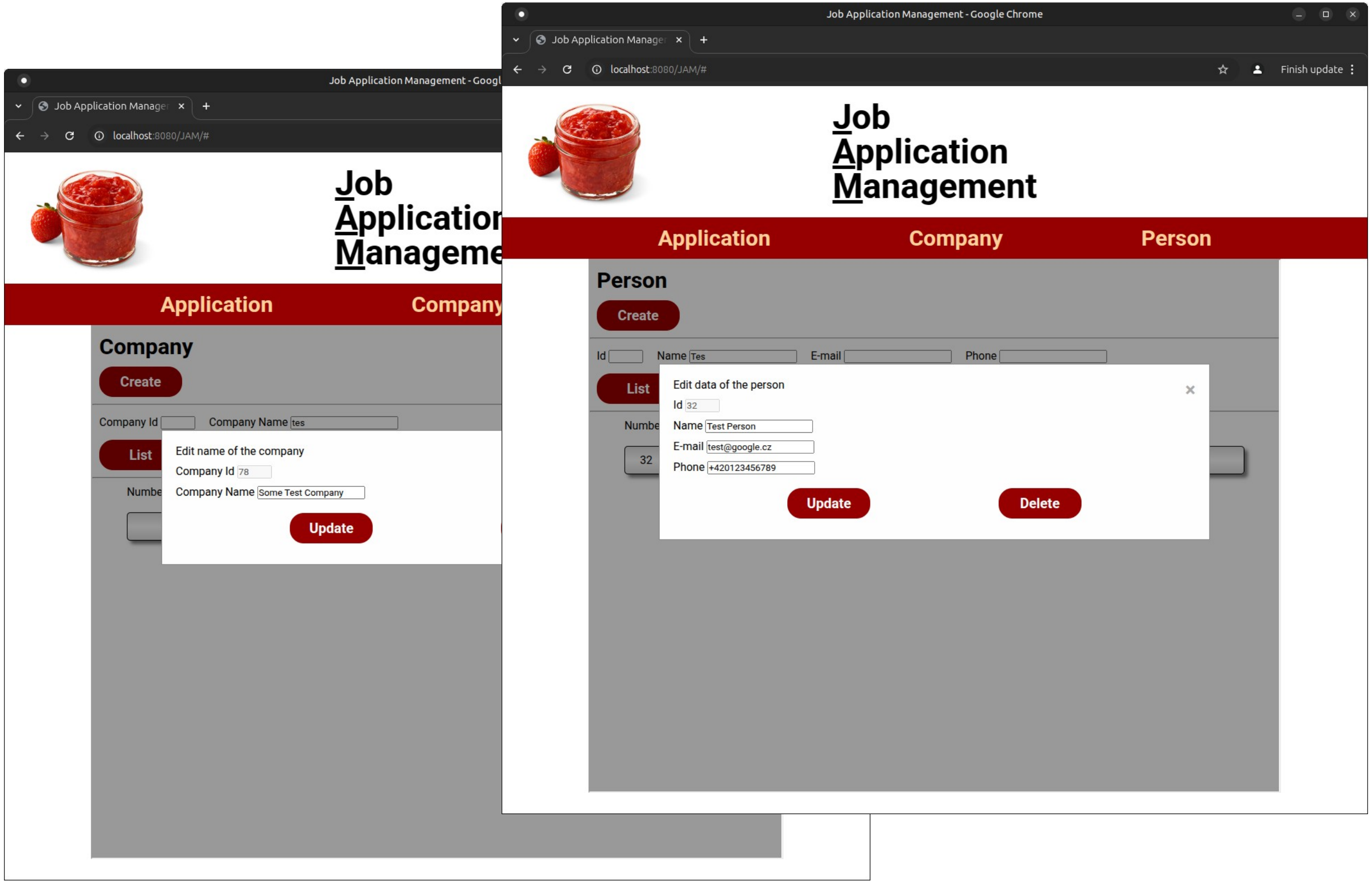
Application details

Application Id 39 Title Test for Autoclose1 Position Id Unknown

Start date 2025-04-15T12:34:57 Last change 2025-04-16T14:46:34 Status Closed

Front-end layer

Same approach for editing an entity using modal like dialog is used for company and person as well



Job Application Management - Googl

Job Application Manage

localhost:8080/JAM/#

Job Application Manageme

Company

Company Id

Company Name tes

Numbe

Edit name of the company

Company Id 78

Company Name Some Test Company

Job Application Management - Google Chrome

Job Application Manage

localhost:8080/JAM/#

Finish update

Job Application Management

Person

Id

Name Tes

E-mail

Phone

Numbe

32

Edit data of the person

Id 32

Name Test Person

E-mail test@google.cz

Phone +420123456789

JAM

Source code available on https://github.com/radimSed/JAM.git

Other work available on https://github.com/radimSed