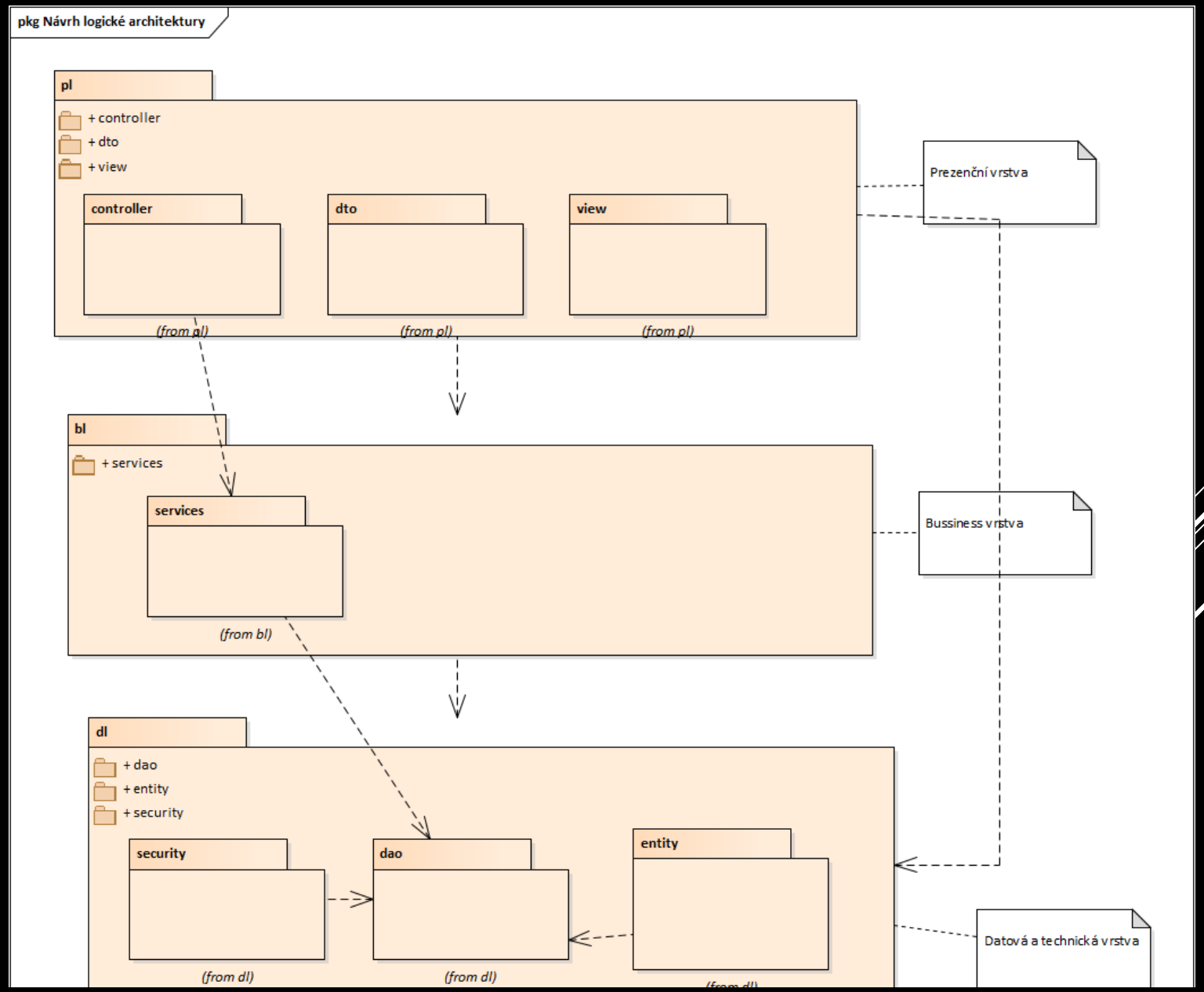
# TATTOOPRO

Autoři:
Jorge Zuniga
Vít Šprachta
Ondřej Cihlář
Michal Patera
Adam Pončák
Michal Seibert

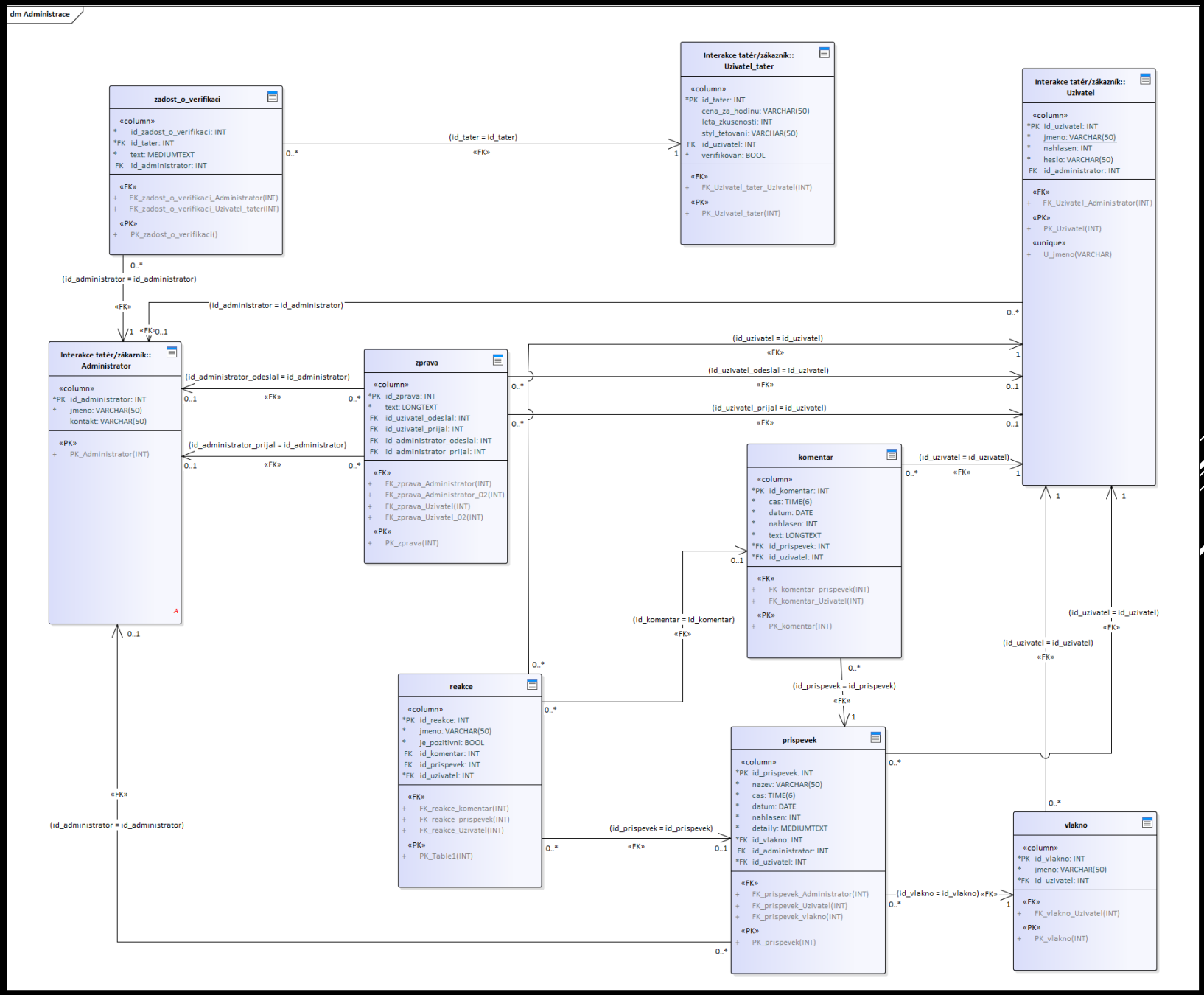
# NÁVRH LOGICKÉ ARCHITEKTURY

3 vrstvá architektura

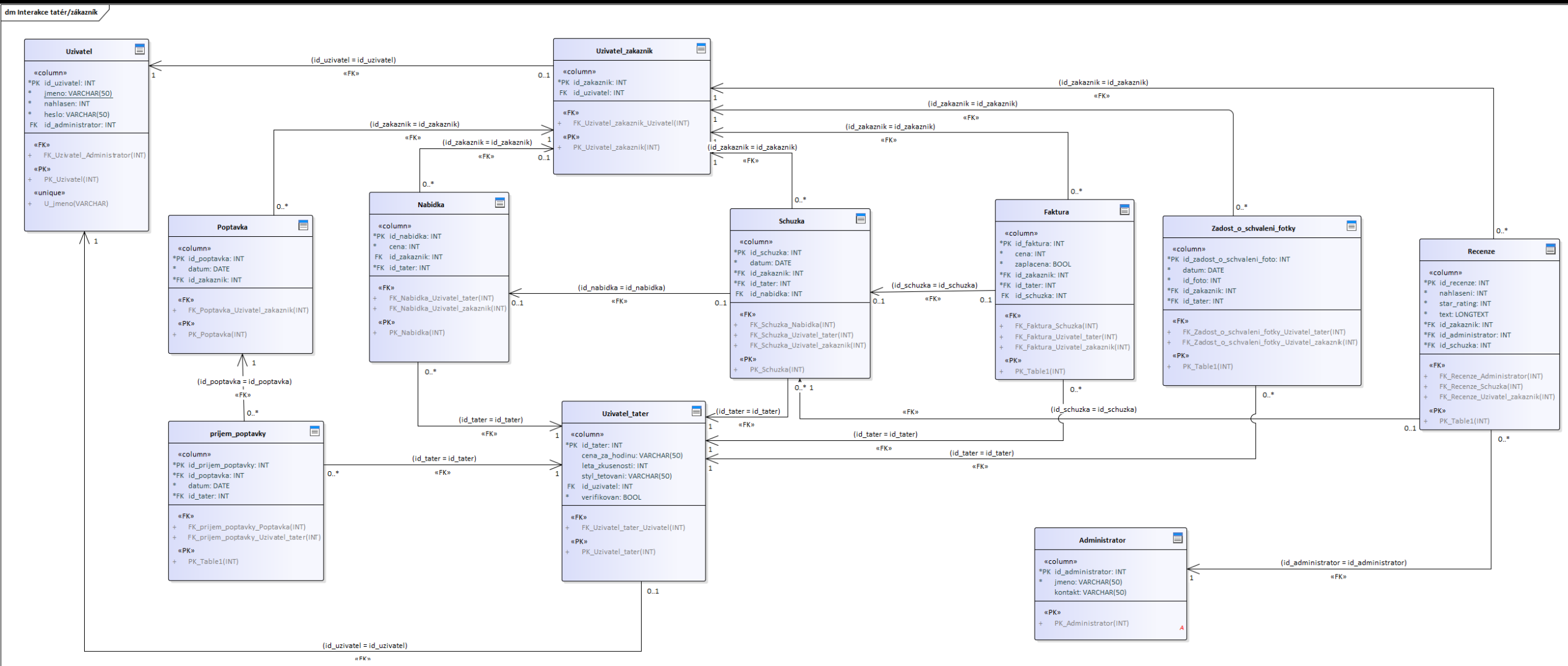


# DATABÁZOVÝ MODEL

# ADMINISTRACE



# INTERAKCE TATÉR/ZÁKAZNÍK



# CUSTOMER DTO

```java
package fit.cvut.si1.semestralka.tattooPro.web.DataTransferObjects;

import fit.cvut.si1.semestralka.tattooPro.data.entityModels.Customer;

/**
 * DTO for customer
 */
public class CustomerDTO extends UserDTO{

    public CustomerDTO(Customer customer) { super(customer.getUsername(),customer.getEmail()); }


    public CustomerDTO(String username, String email) { super(username, email); }
}
```

# PROFILE CONTROLLER

```java
package fit.cvut.sil.semestralka.tattooPro.web;


import ...

/**
 * Controller for handling profiles - loging, editing etc.
 */
@Controller
@RequestMapping(value = "/")
public class profileController {

    @GetMapping(value = "/myprofile")
    public String getCurrentProfile(Model model){

        User user = (User)SecurityContextHolder.getContext().getAuthentication().getPrincipal();

        model.addAttribute( s: "profile", user);
        return "myProfile";
    }

    @GetMapping(value = "/registration")
    public String registrateUser(Model model) { return "registration"; }


    @GetMapping(value = "/login")
    public String loginUser(Model model) { return "login"; }

}
```

# CUSTOMER SERVICE

```java
package fit.cvut.si1.semestralka.tattooPro.service;

import ...

/**
 * Customer service class.
 */
@Service
public class CustomerService {

    @Autowired
    private CustomerDAO customerDAO;

    @Transactional
    public boolean add(Customer customer){
        if(!findById(customer.getUserID()).isEmpty()) return false;
        customerDAO.persist(customer);
        return true;
    }

    @Transactional
    public void update(Customer customer){
        if( findById(customer.getUserID()).isEmpty()) customerDAO.persist(customer);
        else customerDAO.update(customer);
    }

    @Transactional
    public boolean remove(int id){
        List<Customer> toR = findById(id);
        if(toR.isEmpty()) return false;
        customerDAO.remove(toR.get(0));
        return true;
    }
```

# CUSTOMER SERVICE

```
    @Transactional
    public void addAll(List<Customer> customerList){
        for(Customer customer : customerList){
            customerDAO.persist(customer);
        }

    }

    @Transactional(readOnly = true)
    public List<Customer> listAll() { return customerDAO.findAll(); }

    @Transactional(readOnly = true)
    public List<Customer> findById(int id) { return customerDAO.findById(id); }

    @Transactional(readOnly = true)
    public List<Customer> findByUsername(String username) { return  customerDAO.findByUsername(username); }
}
```

# CUSTOMER DAO

```
package fit.cvut.si1.semestralka.tattooPro.data.dataAccessObj;

import ...

@Repository
public class CustomerDAO {
    @PersistenceContext
    private EntityManager em;

    public void persist(Customer customer) { em.persist(customer); }

    public void update(Customer customer) { em.merge(customer); }
    public void remove(Customer customer) { em.remove(customer); }

    public List findAll() { return em.createQuery( s: "SELECT c FROM Customer c").getResultList(); }

    public List findById(int id){
        return em.createQuery( s: "SELECT c FROM Customer c WHERE c.UserID = :uid").setParameter( s: "uid",id).getResultList();
    }
    public List findByUsername(String username){
        return em.createQuery( s: "SELECT c FROM Customer c WHERE c.Username = :uid").setParameter( s: "uid",username).getResultList();
    }

}
```

# USER ENTITY

```java
package fit.cvut.si1.semestralka.tattooPro.data.entityModels;

import javax.persistence.*;

/**
 * User class represents user.
 * Users can be strictly one of two types. Customer OR TattooArtist
 */
@MappedSuperclass
public class User {
    /**
     * UserID
     */
    @Id
    @GeneratedValue
    protected int UserID;
    /**
     * User username.
     */
    @Column(unique = true, nullable = false)
    protected String Username;
    /**
     * User email.
     */
    @Column(unique = true)
    protected String Email;
    /**
     * User password.
     */
    @Column(nullable = false)
    protected String Password;
    /**
     * Number of times the user has been reported.
     */
    protected int NumberOfReports;
```

# USER ENTITY

```
public User(){}

public User(String username, String email, String password, int numberOfReports, int userID){
    Username = username;
    Email = email;
    Password = password;
    NumberOfReports = numberOfReports;
    UserID = userID;
}

public void setUsername(String username) { Username = username; }

public String getUsername() { return Username; }

public void setEmail(String email){Email = email;}

public String getEmail() { return Email; }

public void setPassword(String password) { Password = password; }

public String getPassword() { return Password; }

public void setUserID(int userID) { UserID = userID; }

public int getUserID() { return UserID; }

public void setNumberOfReports(int numberOfReports) { NumberOfReports = numberOfReports; }

public int getNumberOfReports() { return NumberOfReports; }
```

# CUSTOMER ENTITY

```java
package fit.cvut.si1.semestralka.tattooPro.data.entityModels;

import javax.persistence.Entity;

/**
 * Customer entity represents a customer.
 * Every customer is a user.
 * Every customer is NOT a tattoo artist.
 */
@Entity
public class Customer extends User {
    /**
     * Empty constructor.
     */
    public Customer(){}

    /**
     * Constructor with parameters
     * @param username Username.
     * @param numberOfReports Number of reports.
     * @param userID user ID.
     */
    public Customer(String username, String password, int numberOfReports, int userID){
        Username = username;
        Password = password;
        NumberOfReports = numberOfReports;
        UserID = userID;
    }
}
```