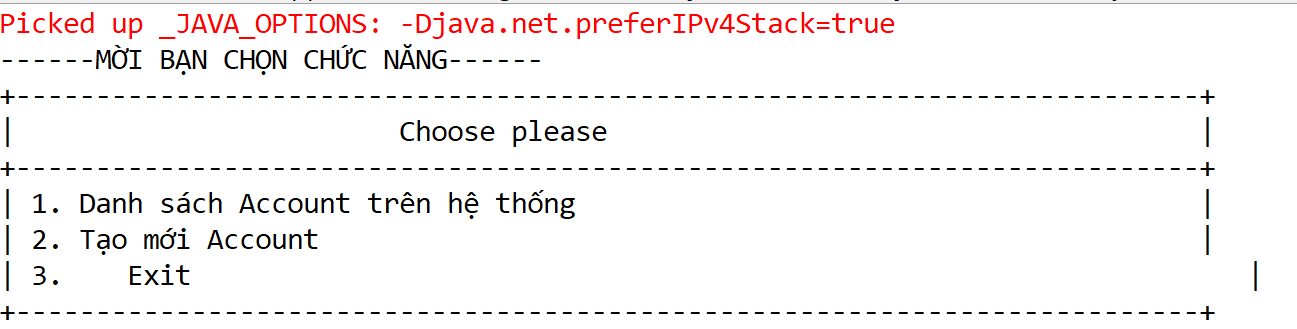
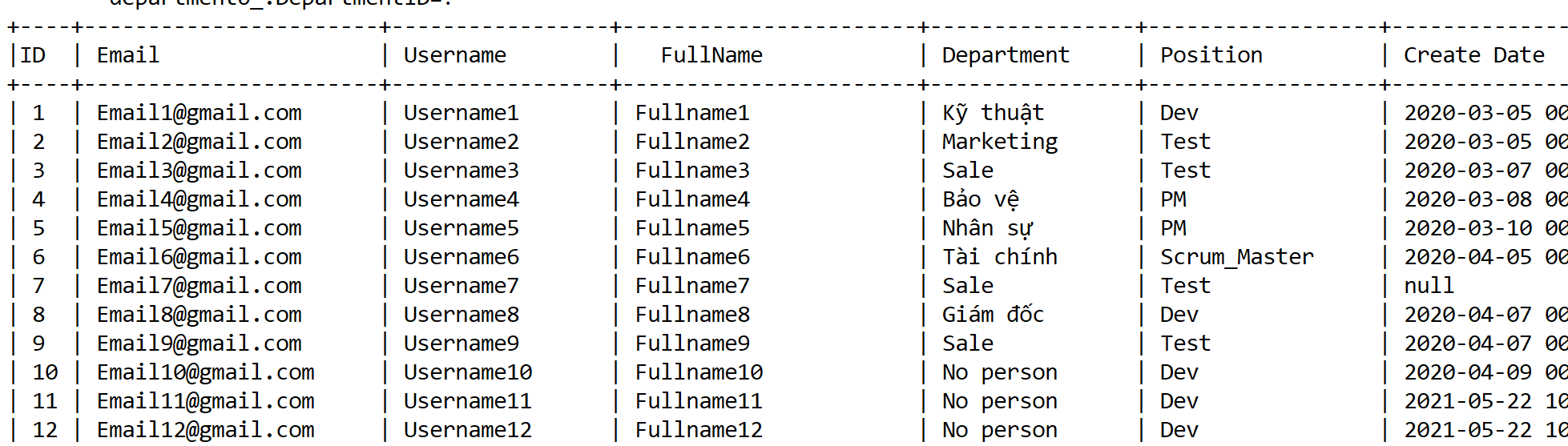
# Lab 4: Demo BeanValidation\_Custom.

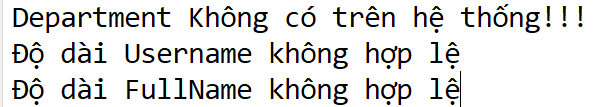
# **Yêu cầu:**

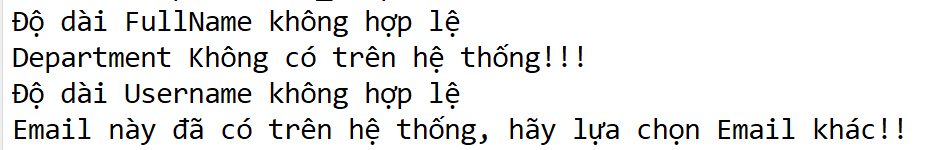
Sử dụng Bean Validation dạng Custom để tạo các Anotation kiểm tra khi tạo mới 1 Account trên hệ thống thì cần check xem Email đã có trên ht hay chưa, Department này có trên hệ thống hay không.

# **Giao diện chương trình**

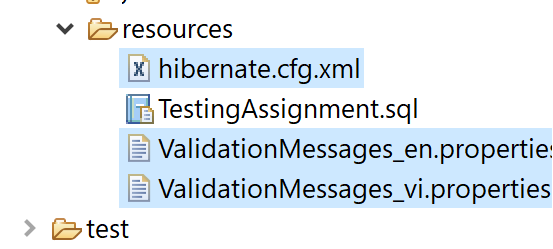


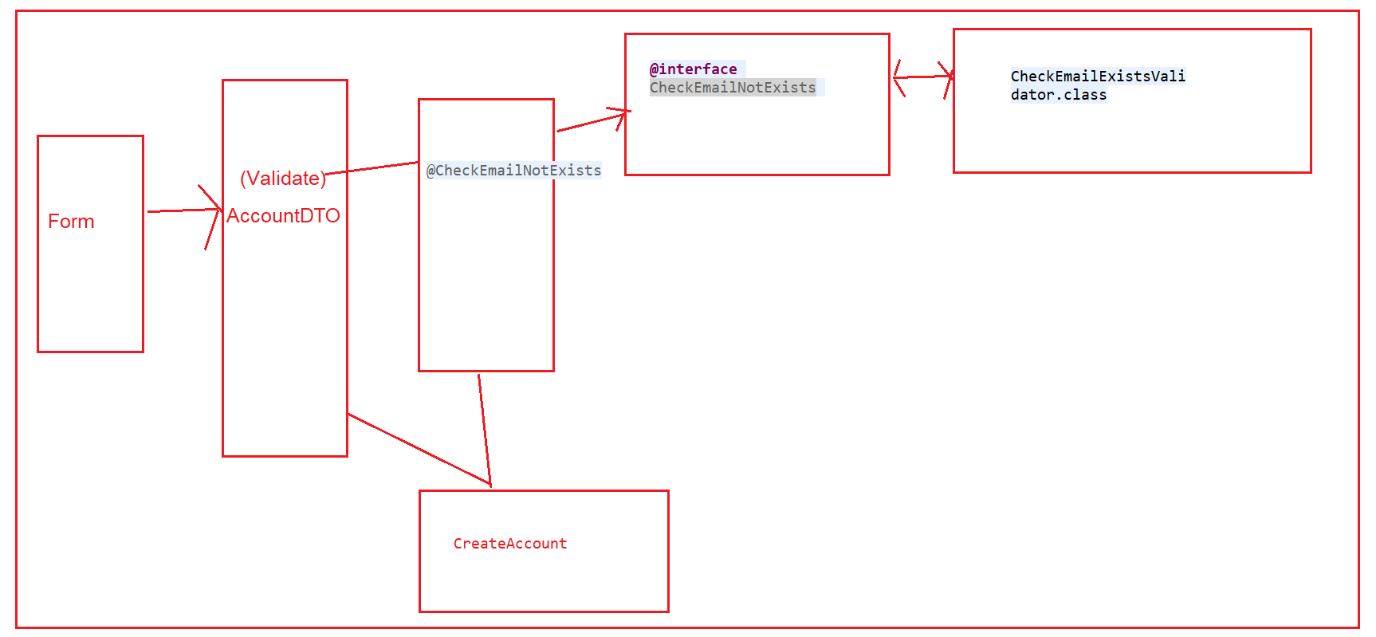








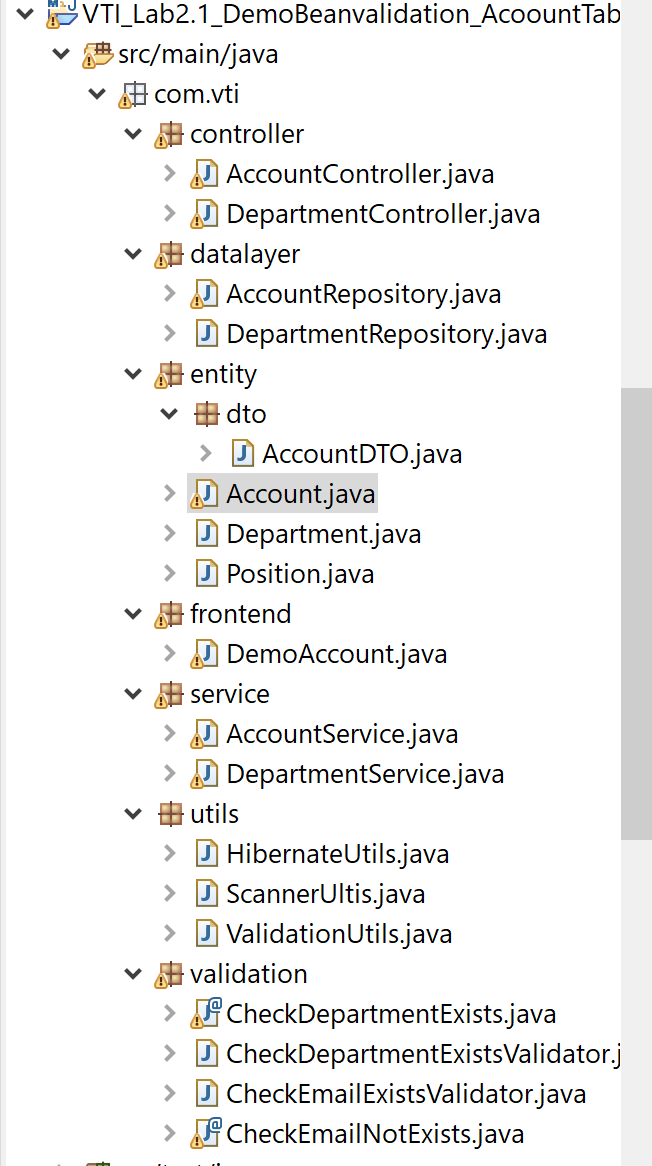


# **Tạo** khung **chương trình:**

Tạo mới 1 project maven: VTI\_Lab2.1\_DemoBeanvalidation\_AcoountTable.

Tạo mới các package và class như dưới:

Package: com.vti.entity, com.vti.frontend, com.vti.controller, com.vti.service, com.vti.datalayer, com.vti.entity.dto, com.vti.utils, com.vti.validation



# **Thêm các cấu hình cho file: pom.xml**

|  |
| --- |
| <!-- https://mvnrepository.com/artifact/javax.validation/validation-api -->  <dependency>  <groupId>javax.validation</groupId>  <artifactId>validation-api</artifactId>  <version>2.0.1.Final</version>  </dependency>  <!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-validator -->  <dependency>  <groupId>org.hibernate</groupId>  <artifactId>hibernate-validator</artifactId>  <version>6.1.5.Final</version>  </dependency>  <!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-validator-annotation-processor -->  <dependency>  <groupId>org.hibernate</groupId>  <artifactId>hibernate-validator-annotation-processor</artifactId>  <version>6.1.5.Final</version>  </dependency>  <!-- https://mvnrepository.com/artifact/javax.el/javax.el-api -->  <dependency>  <groupId>javax.el</groupId>  <artifactId>javax.el-api</artifactId>  <version>3.0.0</version>  </dependency>  <!-- https://mvnrepository.com/artifact/org.glassfish.web/javax.el -->  <dependency>  <groupId>org.glassfish.web</groupId>  <artifactId>javax.el</artifactId>  <version>2.2.6</version>  </dependency>  <!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-core -->  <dependency>  <groupId>org.hibernate</groupId>  <artifactId>hibernate-core</artifactId>  <version>5.4.17.Final</version>  </dependency>  <!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->  <dependency>  <groupId>mysql</groupId>  <artifactId>mysql-connector-java</artifactId>  <version>8.0.20</version>  </dependency>  <dependency>  <groupId>com.github.v-ladynev</groupId>  <artifactId>fluent-hibernate-core</artifactId>  <version>0.3.1</version>  </dependency> |

****

# **Tạo Class HibernateUtils trong package com.vti. utils:**

|  |
| --- |
| **package** com.vti.utils;  **import** org.hibernate.Session;  **import** org.hibernate.SessionFactory;  **import** org.hibernate.boot.registry.StandardServiceRegistryBuilder;  **import** org.hibernate.cfg.Configuration;  **import** org.hibernate.service.ServiceRegistry;  **import** com.vti.entity.Account;  **import** com.vti.entity.Department;  **import** com.vti.entity.Position;  **public** **class** HibernateUtils {  **private** **static** HibernateUtils *instance*;  **private** Configuration configuration;  **private** SessionFactory sessionFactory;  **public** **static** HibernateUtils getInstance() {  **if** (**null** == *instance*) {  *instance* = **new** HibernateUtils();  }  **return** *instance*;  }  **private** HibernateUtils() {  configure();  }  **private** **void** configure() {  // load configuration  configuration = **new** Configuration();  configuration.configure("hibernate.cfg.xml");  // add entity  configuration.addAnnotatedClass(Account.**class**);  configuration.addAnnotatedClass(Department.**class**);  configuration.addAnnotatedClass(Position.**class**);  }  **private** SessionFactory buildSessionFactory() {  **if** (**null** == sessionFactory || sessionFactory.isClosed()) {  ServiceRegistry serviceRegistry = **new** StandardServiceRegistryBuilder()  .applySettings(configuration.getProperties()).build();  sessionFactory = configuration.buildSessionFactory(serviceRegistry);  }  **return** sessionFactory;  }  **public** **void** closeFactory() {  **if** (**null** != sessionFactory && sessionFactory.isOpen()) {  sessionFactory.close();  }  }  **public** Session openSession() {  buildSessionFactory();  **return** sessionFactory.openSession();  }  } |



# **Tạo Class Student trong package com.vti. entity:**

|  |
| --- |
| **package** com.vti.entity;  **import** javax.validation.constraints.Email;  **import** javax.validation.constraints.Max;  **import** javax.validation.constraints.Min;  **public** **class** Student {  @Min(value = 1, message = "ID không hợp lệ!!")  // @Positive  // @Size(min = 1, max = 3, message = "ID không hợp lệ")  **private** **int** id;  @Min(value = 6, message = "Tên không hợp lệ!")  @Max(value = 12, message = "Tên không hợp lệ!")  **private** String name;  @Email(message = "Thông tin email không hợp lệ!")  **private** String email;  @Min(value = 18, message = "Tuổi không hợp lệ!")  **private** **int** age;  /\*\*  \* **@return** the id  \*/  **public** **int** getId() {  **return** id;  }  /\*\*  \* **@param** id the id to set  \*/  **public** **void** setId(**int** id) {  **this**.id = id;  }  /\*\*  \* **@return** the name  \*/  **public** String getName() {  **return** name;  }  /\*\*  \* **@param** name the name to set  \*/  **public** **void** setName(String name) {  **this**.name = name;  }  /\*\*  \* **@return** the email  \*/  **public** String getEmail() {  **return** email;  }  /\*\*  \* **@param** email the email to set  \*/  **public** **void** setEmail(String email) {  **this**.email = email;  }  /\*\*  \* **@return** the age  \*/  **public** **int** getAge() {  **return** age;  }  /\*\*  \* **@param** age the age to set  \*/  **public** **void** setAge(**int** age) {  **this**.age = age;  }  **public** Student(**int** id, String name, String email, **int** age) {  **super**();  **this**.id = id;  **this**.name = name;  **this**.email = email;  **this**.age = age;  }  **public** Student() {  // **TODO** Auto-generated constructor stub  }  @Override  **public** String toString() {  **return** "Student [id=" + id + ", name=" + name + ", email=" + email + ", age=" + age + "]";  }  } |



# **Tạo Class ScannerUltis trong package com.vti. utils:**

|  |
| --- |
| **package** com.vti.utils;  **import** java.text.SimpleDateFormat;  **import** java.time.LocalDate;  **import** java.util.Scanner;  **public** **class** ScannerUltis {  **private** **static** Scanner *sc* = **new** Scanner(System.***in***);  **public** **static** **int** inputInt() {  **while** (**true**) {  **try** {  **return** Integer.*parseInt*(*sc*.next().trim());  } **catch** (Exception e) {  System.***err***.println("Nhập lại:");  }  }  }  **public** **static** **int** inputIntPositive() {  **while** (**true**) {  **try** {  **int** intPositive = Integer.*parseInt*(*sc*.next());  **if** (intPositive >= 0) {  **return** intPositive;  } **else** {  System.***err***.println("Nhập lại:");  }  } **catch** (Exception e) {  System.***err***.println("Nhập lại:");  }  }  }  **public** **static** Float inputFloat() {  **while** (**true**) {  **try** {  **return** Float.*parseFloat*(*sc*.next());  } **catch** (Exception e) {  System.***err***.println("Nhập lại:");  }  }  }  **public** **static** Double inputDouble() {  **while** (**true**) {  **try** {  **return** Double.*parseDouble*(*sc*.next());  } **catch** (Exception e) {  System.***err***.println("Nhập lại:");  }  }  }  **public** **static** String inputString() {  **while** (**true**) {  String string = *sc*.next().trim();  **if** (!string.isEmpty()) {  **return** string;  } **else** {  System.***err***.println("Nhập lại:");  }  }  }  **public** **static** LocalDate inputLocalDate() {  System.***out***.println("Nhập theo định dạng yyyy-MM-dd");  SimpleDateFormat format = **new** SimpleDateFormat("yyyy-MM-dd");  **while** (**true**) {  String localdate = *sc*.next().trim();  **try** {  **if** (format.parse(localdate) != **null**) {  LocalDate lc = LocalDate.*parse*(localdate);  **return** lc;  }  } **catch** (Exception e) {  System.***err***.println("Nhập lại:");  }  }  }  **public** **static** String inputEmail() {  **while** (**true**) {  String email = ScannerUltis.*inputString*();  **if** (email == **null** || !email.contains("@")) {  System.***out***.print("Nhập lại: ");  } **else** {  **return** email;  }  }  }  **public** **static** **int** inputFunction(**int** a, **int** b, String errorMessage) {  **while** (**true**) {  **int** number = ScannerUltis.*inputInt*();  **if** (number >= a && number <= b) {  **return** number;  } **else** {  System.***err***.println(errorMessage);  }  }  }  **public** **static** String inputPassword() {  **while** (**true**) {  String password = ScannerUltis.*inputString*();  **if** (password.length() < 6 || password.length() > 12) {  System.***out***.print("Nhập lại: ");  **continue**;  }  **boolean** hasAtLeast1Character = **false**;  **for** (**int** i = 0; i < password.length(); i++) {  **if** (Character.*isUpperCase*(password.charAt(i)) == **true**) {  hasAtLeast1Character = **true**;  **break**;  }  }  **if** (hasAtLeast1Character == **true**) {  **return** password;  } **else** {  System.***out***.print("Mời bạn nhập lại password: ");  }  }  }  **public** **static** String inputPhoneNumber() {  **while** (**true**) {  String number = ScannerUltis.*inputString*();  **if** (number.length() > 12 || number.length() < 9) {  **continue**;  }  **if** (number.charAt(0) != '0') {  **continue**;  }  **boolean** isNumber = **true**;  **for** (**int** i = 0; i < number.length(); i++) {  **if** (Character.*isDigit*(number.charAt(i)) == **false**) {  isNumber = **false**;  **break**;  }  }  **if** (isNumber == **true**) {  **return** number;  } **else** {  System.***out***.print("Nhập lại: ");  }  }  }  } |



# **Tạo Class ValidationUtils trong package com.vti. utils:**

|  |
| --- |
| **package** com.vti.utils;  **import** java.util.Set;  **import** javax.validation.ConstraintViolation;  **import** javax.validation.Validation;  **import** javax.validation.Validator;  **import** javax.validation.ValidatorFactory;  **public** **class** ValidationUtils {  **public** **static** <T> **boolean** validate(T dto) {  ValidatorFactory validatorFactory = Validation.*buildDefaultValidatorFactory*();  Validator validator = validatorFactory.getValidator();  Set<ConstraintViolation<T>> constraintViolations = validator.validate(dto);  **if** (**null** == constraintViolations || constraintViolations.size() == 0) {  **return** **true**;  }  **for** (ConstraintViolation<T> violation : constraintViolations) {  System.***out***.println(violation.getMessage());  }  **return** **false**;  }  } |



# **Tạo Class Account trong package com.vti. entity:**

|  |
| --- |
| **package** com.vti.entity;  **import** java.io.Serializable;  **import** java.util.Date;  **import** javax.persistence.Column;  **import** javax.persistence.Entity;  **import** javax.persistence.GeneratedValue;  **import** javax.persistence.GenerationType;  **import** javax.persistence.Id;  **import** javax.persistence.Inheritance;  **import** javax.persistence.InheritanceType;  **import** javax.persistence.JoinColumn;  **import** javax.persistence.ManyToOne;  **import** javax.persistence.Table;  **import** javax.persistence.Temporal;  **import** javax.persistence.TemporalType;  **import** org.hibernate.annotations.CreationTimestamp;  **import** org.hibernate.annotations.Formula;  @Entity  @Table(name = "`Account`", catalog = "TestingSystem")  **public** **class** Account **implements** Serializable {  @Column(name = "AccountID")  @Id  @GeneratedValue(strategy = GenerationType.IDENTITY)  **private** **short** id;  @Column(name = "Email", length = 50, nullable = **false**, unique = **true**, updatable = **false**)  **private** String email;  @Column(name = "Username", length = 50, nullable = **false**, unique = **true**, updatable = **false**)  **private** String username;  @Column(name = "FullName", length = 50, nullable = **false**)  **private** String fullName;  @ManyToOne  @JoinColumn(name = "DepartmentID")  **private** Department department;  @ManyToOne  @JoinColumn(name = "PositionID")  **private** Position position;  @Column(name = "CreateDate")  @Temporal(TemporalType.TIMESTAMP)  @CreationTimestamp  **private** Date createDate;  **public** Account() {  **super**();  }  **public** Account(String email, String username, String fullName, Department department) {  **super**();  **this**.email = email;  **this**.username = username;  **this**.fullName = fullName;  **this**.department = department;  }  /\*\*  \* **@return** the id  \*/  **public** **short** getId() {  **return** id;  }  /\*\*  \* **@param** id the id to set  \*/  **public** **void** setId(**short** id) {  **this**.id = id;  }  /\*\*  \* **@return** the email  \*/  **public** String getEmail() {  **return** email;  }  /\*\*  \* **@param** email the email to set  \*/  **public** **void** setEmail(String email) {  **this**.email = email;  }  /\*\*  \* **@return** the username  \*/  **public** String getUsername() {  **return** username;  }  /\*\*  \* **@param** username the username to set  \*/  **public** **void** setUsername(String username) {  **this**.username = username;  }  /\*\*  \* **@return** the fullName  \*/  **public** String getFullName() {  **return** fullName;  }  /\*\*  \* **@param** fullName the fullName to set  \*/  **public** **void** setFullName(String fullName) {  **this**.fullName = fullName;  }  /\*\*  \* **@return** the department  \*/  **public** Department getDepartment() {  **return** department;  }  /\*\*  \* **@param** department the department to set  \*/  **public** **void** setDepartment(Department department) {  **this**.department = department;  }  /\*\*  \* **@return** the position  \*/  **public** Position getPosition() {  **return** position;  }  /\*\*  \* **@param** position the position to set  \*/  **public** **void** setPosition(Position position) {  **this**.position = position;  }  /\*\*  \* **@return** the createDate  \*/  **public** Date getCreateDate() {  **return** createDate;  }  /\*\*  \* **@param** createDate the createDate to set  \*/  **public** **void** setCreateDate(Date createDate) {  **this**.createDate = createDate;  }  } |



# **Tạo Class Department trong package com.vti. entity:**

|  |
| --- |
| **package** com.vti.entity;  **import** java.io.Serializable;  **import** java.util.List;  **import** javax.persistence.Column;  **import** javax.persistence.Entity;  **import** javax.persistence.FetchType;  **import** javax.persistence.GeneratedValue;  **import** javax.persistence.GenerationType;  **import** javax.persistence.Id;  **import** javax.persistence.OneToMany;  **import** javax.persistence.Table;  @SuppressWarnings("serial")  @Entity  @Table(name = "Department", catalog = "TestingSystem")  **public** **class** Department **implements** Serializable {  @Column(name = "DepartmentID")  @Id  @GeneratedValue(strategy = GenerationType.***IDENTITY***)  **private** **short** id;  @Column(name = "DepartmentName", length = 30, nullable = **false**, unique = **true**)  **private** String name;  @OneToMany(mappedBy = "department", fetch = FetchType.***EAGER***)  **private** List<Account> account;  **public** Department() {  **super**();  }  /\*\*  \* **@return** the id  \*/  **public** **short** getId() {  **return** id;  }  /\*\*  \* **@param** id the id to set  \*/  **public** **void** setId(**short** id) {  **this**.id = id;  }  /\*\*  \* **@return** the name  \*/  **public** String getName() {  **return** name;  }  /\*\*  \* **@param** name the name to set  \*/  **public** **void** setName(String name) {  **this**.name = name;  }  @Override  **public** String toString() {  **return** "Department [id=" + id + ", name=" + name + "]";  }  } |



# **Tạo Class Position trong package com.vti. entity:**

|  |
| --- |
| **package** com.vti.entity;  **import** java.io.Serializable;  **import** java.util.List;  **import** javax.persistence.Column;  **import** javax.persistence.Entity;  **import** javax.persistence.EnumType;  **import** javax.persistence.Enumerated;  **import** javax.persistence.GeneratedValue;  **import** javax.persistence.GenerationType;  **import** javax.persistence.Id;  **import** javax.persistence.OneToMany;  **import** javax.persistence.Table;  @Entity  @Table(name = "Position", catalog = "TestingSystem")  **public** **class** Position **implements** Serializable {  **private** **static** **final** **long** serialVersionUID = 1L;  @Column(name = "PositionID")  @Id  @GeneratedValue(strategy = GenerationType.IDENTITY)  **private** **short** id;  @Column(name = "PositionName", nullable = **false**, unique = **true**)  @Enumerated(EnumType.STRING)  **private** PositionName name;  **public** **enum** PositionName {  Dev, Test, Scrum\_Master, PM  }  @OneToMany(mappedBy = "position")  List<Account> accounts;  **public** Position() {  **super**();  }  /\*\*  \* **@return** the id  \*/  **public** **short** getId() {  **return** id;  }  /\*\*  \* **@param** id the id to set  \*/  **public** **void** setId(**short** id) {  **this**.id = id;  }  /\*\*  \* **@return** the name  \*/  **public** PositionName getName() {  **return** name;  }  /\*\*  \* **@param** name the name to set  \*/  **public** **void** setName(PositionName name) {  **this**.name = name;  }  /\*\*  \* **@return** the accounts  \*/  **public** List<Account> getAccounts() {  **return** accounts;  }  /\*\*  \* **@param** accounts the accounts to set  \*/  **public** **void** setAccounts(List<Account> accounts) {  **this**.accounts = accounts;  }  } |



# **Tạo Class AccountDTO trong package dto:**

|  |
| --- |
| **package** com.vti.entity.dto;  **import** javax.validation.constraints.Email;  **import** javax.validation.constraints.NotBlank;  **import** javax.validation.constraints.Positive;  **import** org.hibernate.validator.constraints.Length;  **import** com.vti.datalayer.DepartmentRepository;  **import** com.vti.entity.Account;  **import** com.vti.entity.Department;  **import** com.vti.validation.CheckDepartmentExists;  **import** com.vti.validation.CheckEmailNotExists;  **public** **class** AccountDTO {  @NotBlank(message = "Email không được để trống")  @Length(min = 6, max = 50, message = "Độ dài email không hợp lệ")  @Email(message = "Format Email không hợp lệ")  @CheckEmailNotExists(message = "Email này đã có trên hệ thống, hãy lựa chọn Email khác!!")  **private** String email;  @NotBlank(message = "Username không được để trống")  @Length(min = 6, max = 50, message = "Độ dài Username không hợp lệ")  **private** String username;  @NotBlank(message = "FullName không được để trống")  @Length(min = 6, max = 50, message = "Độ dài FullName không hợp lệ")  **private** String fullname;  @Positive(message = "Lựa chọn ID là số dương")  @CheckDepartmentExists  **private** **int** departmentID;  **public** AccountDTO(String email, String username, String fullname) {  **super**();  **this**.email = email;  **this**.username = username;  **this**.fullname = fullname;  }  **public** AccountDTO() {  // **TODO** Auto-generated constructor stub  }  /\*\*  \* **@return** the email  \*/  **public** String getEmail() {  **return** email;  }  /\*\*  \* **@param** email the email to set  \*/  **public** **void** setEmail(String email) {  **this**.email = email;  }  /\*\*  \* **@return** the username  \*/  **public** String getUsername() {  **return** username;  }  /\*\*  \* **@param** username the username to set  \*/  **public** **void** setUsername(String username) {  **this**.username = username;  }  /\*\*  \* **@return** the fullname  \*/  **public** String getFullname() {  **return** fullname;  }  /\*\*  \* **@param** fullname the fullname to set  \*/  **public** **void** setFullname(String fullname) {  **this**.fullname = fullname;  }  /\*\*  \* **@return** the departmentID  \*/  **public** **int** getDepartmentID() {  **return** departmentID;  }  /\*\*  \* **@param** departmentID the departmentID to set  \*/  **public** **void** setDepartmentID(**int** departmentID) {  **this**.departmentID = departmentID;  }  **public** Account toEntity() {  DepartmentRepository departmentRepository = **new** DepartmentRepository();  Department department = departmentRepository.getDepartmentByID((**short**) departmentID);  **return** **new** Account(email, username, fullname, department);  }  } |

****

# **Tạo Class AccountRepository trong package datalayer:**

|  |
| --- |
| **package** com.vti.datalayer;  **import** java.util.List;  **import** org.hibernate.Session;  **import** org.hibernate.query.Query;  **import** com.vti.entity.Account;  **import** com.vti.utils.HibernateUtils;  **public** **class** AccountRepository {  **private** HibernateUtils hibernateUtils;  **public** AccountRepository() {  hibernateUtils = HibernateUtils.getInstance();  }  @SuppressWarnings("unchecked")  **public** List<Account> getAllAccount() {  Session session = **null**;  **try** {  // get session  session = hibernateUtils.openSession();  // create hql query  Query<Account> query = session.createQuery("FROM Account");  **return** query.list();  } **finally** {  **if** (session != **null**) {  session.close();  }  }  }  **public** **void** createAccount(Account Account) {  Session session = **null**;  **try** {  // get session  session = hibernateUtils.openSession();  session.beginTransaction();  // create  session.save(Account);  session.getTransaction().commit();  } **finally** {  **if** (session != **null**) {  session.close();  }  }  }  **public** **boolean** isAccountExistsByEmail(String email) {  // get account  Account account = getAccountByEmail(email);  // return result  **if** (account == **null**) {  **return** **false**;  }  **return** **true**;  }  **public** Account getAccountByEmail(String email) {  Session session = **null**;  **try** {  // get session  session = hibernateUtils.openSession();  // create hql query  Query<Account> query = session.createQuery("FROM Account WHERE email = :emailParameter");  // set parameter  query.setParameter("emailParameter", email);  // get result  Account account = query.uniqueResult();  **return** account;  } **finally** {  **if** (session != **null**) {  session.close();  }  }  }  } |

****

# **Tạo Class DepartmentRepository trong package datalayer:**

|  |
| --- |
| **package** com.vti.datalayer;  **import** java.util.List;  **import** org.hibernate.Session;  **import** org.hibernate.query.Query;  **import** com.vti.entity.Department;  **import** com.vti.utils.HibernateUtils;  **public** **class** DepartmentRepository {  **private** HibernateUtils hibernateUtils;  **public** DepartmentRepository() {  hibernateUtils = HibernateUtils.getInstance();  }  @SuppressWarnings("unchecked")  **public** List<Department> getAllDepartment() {  Session session = **null**;  **try** {  // get session  session = hibernateUtils.openSession();  // create hql query  Query<Department> query = session.createQuery("FROM Department order by id");  **return** query.list();  } **finally** {  **if** (session != **null**) {  session.close();  }  }  }  **public** Department getDepartmentByID(**short** id) {  Session session = **null**;  **try** {  session = hibernateUtils.openSession();  Department department = session.get(Department.**class**, id);  **return** department;  } **finally** {  **if** (session != **null**) {  session.close();  }  }  }  **public** **boolean** isDepartmentExistsByID(**short** id) {  // get department  Department department = getDepartmentByID(id);  // return result  **if** (department == **null**) {  **return** **false**;  }  **return** **true**;  }  } |

****

# **Tạo Class DepartmentService trong package service:**

|  |
| --- |
| **package** com.vti.service;  **import** java.util.List;  **import** com.vti.datalayer.AccountRepository;  **import** com.vti.datalayer.DepartmentRepository;  **import** com.vti.entity.Account;  **import** com.vti.entity.Department;  **import** com.vti.entity.dto.AccountDTO;  **public** **class** DepartmentService {  **private** DepartmentRepository DepRepository;  **public** DepartmentService() {  DepRepository = **new** DepartmentRepository();  }  @SuppressWarnings("unchecked")  **public** List<Department> getAllDepartmentAccounts() {  **return** DepRepository.getAllDepartment();  }  } |

****

# **Tạo Class AccountService trong package service:**

|  |
| --- |
| **package** com.vti.service;  **import** java.util.List;  **import** com.vti.datalayer.AccountRepository;  **import** com.vti.entity.Account;  **import** com.vti.entity.dto.AccountDTO;  **public** **class** AccountService {  **private** AccountRepository accRepository;  **public** AccountService() {  accRepository = **new** AccountRepository();  }  @SuppressWarnings("unchecked")  **public** List<Account> getAllAccount() {  **return** accRepository.getAllAccount();  }  **public** **void** createAccount(AccountDTO accDTO) {  Account entity = accDTO.toEntity();  accRepository.createAccount(entity);  }    **public** Account getAccountByEmail(String email) {  **return** accRepository.getAccountByEmail(email);  }  **public** **boolean** isAccountExistsByEmail(String email) {  **return** accRepository.isAccountExistsByEmail(email);  }  } |

****

# **Tạo Class AccountController trong package controller:**

|  |
| --- |
| **package** com.vti.controller;  **import** java.util.List;  **import** java.util.Locale;  **import** javax.validation.Valid;  **import** com.vti.entity.Account;  **import** com.vti.entity.dto.AccountDTO;  **import** com.vti.service.AccountService;  **import** com.vti.utils.ValidationUtils;  **public** **class** AccountController {  **private** AccountService accService;  **public** AccountController() {  accService = **new** AccountService();  }  @SuppressWarnings("unchecked")  **public** List<Account> getAllAccount() {  **return** accService.getAllAccount();  }  **public** **void** createAccount(@Valid AccountDTO accDTO) {  **if** (ValidationUtils.*validate*(accDTO)) {  accService.createAccount(accDTO);  }  }  } |

****

# **Tạo Class DepartmentController trong package controller:**

|  |
| --- |
| **package** com.vti.controller;  **import** java.util.List;  **import** java.util.Locale;  **import** javax.validation.Valid;  **import** com.vti.entity.Account;  **import** com.vti.entity.Department;  **import** com.vti.entity.dto.AccountDTO;  **import** com.vti.service.AccountService;  **import** com.vti.service.DepartmentService;  **import** com.vti.utils.ValidationUtils;  **public** **class** DepartmentController {  **private** DepartmentService depService;  **public** DepartmentController() {  depService = **new** DepartmentService();  }  @SuppressWarnings("unchecked")  **public** List<Department> getAllDepartments() {  **return** depService.getAllDepartmentAccounts();  }  } |

****

# **Tạo Class DemoAccount trong package frontend:**

|  |
| --- |
| **package** com.vti.frontend;  **import** java.util.List;  **import** java.util.Scanner;  **import** com.vti.controller.AccountController;  **import** com.vti.controller.DepartmentController;  **import** com.vti.entity.Account;  **import** com.vti.entity.Department;  **import** com.vti.entity.dto.AccountDTO;  **import** com.vti.utils.ScannerUltis;  **public** **class** DemoAccount {  **public** **static** **void** main(String[] args) {  **while** (**true**) {  System.out.println("------MỜI BẠN CHỌN CHỨC NĂNG------");  String leftAlignFormat = "| %-72s |%n";  System.out.format("+--------------------------------------------------------------------------+%n");  System.out.format("| Choose please |%n");  System.out.format("+--------------------------------------------------------------------------+%n");  System.out.format(leftAlignFormat, "1. Danh sách Account trên hệ thống");  System.out.format(leftAlignFormat, "2. Tạo mới Account");  System.out.format(leftAlignFormat, "3. Exit");  System.out.format("+--------------------------------------------------------------------------+%n");  **switch** (ScannerUltis.inputIntPositive()) {  **case** 1:  getAllAccount();  **break**;  **case** 2:  createAccount();  **break**;  **case** 3:  **return**;  **default**:  System.out.println("Nhập lại:");  **break**;  }  }  }  **private** **static** **void** createAccount() {  AccountController controller = **new** AccountController();  Scanner scanner = **new** Scanner(System.in);  System.out.println("\n\n\*\*\*\*\*\*\*\*\*\*\*CREATE ACCOUNT\*\*\*\*\*\*\*\*\*\*\*");  AccountDTO acc = **new** AccountDTO();  System.out.println("Nhập vào Email: ");  acc.setEmail(scanner.next());  System.out.println("Nhập vào UserName: ");  acc.setUsername(scanner.next());  System.out.println("Nhập vào FullName: : ");  acc.setFullname(scanner.next());  System.out.println("Hãy chọn phòng nhân viên: ");  **int** depId = getDep();  acc.setDepartmentID(depId);  controller.createAccount(acc);  // getAllAccount();  }  **private** **static** **int** getDep() {  Scanner scanner = **new** Scanner(System.in);  DepartmentController depController = **new** DepartmentController();  List<Department> listDep = depController.getAllDepartments();  String leftAlignFormat = "| %-6d | %-21s |%n";  System.out.format("+--------+-----------------------+%n");  System.out.format("| ID | Depament Name |%n");  System.out.format("+--------+-----------------------+%n");  **for** (Department department : listDep) {  System.out.format(leftAlignFormat, department.getId(), department.getName());  }  System.out.format("+--------+-----------------------+%n");  System.out.println("Chọn phòng theo ID:");  **int** chooseDep = scanner.nextInt();  **return** chooseDep;  }  **private** **static** **void** getAllAccount() {  System.out.println("Danh sách Account trên hệ thống");  AccountController accController = **new** AccountController();  List<Account> listAcc = accController.getAllAccount();  String leftAlignFormat = "| %-2d | %-21s | %-15s | %-21s | %-14s | %-16s | %-16s | %n";  System.out.format(  "+----+-----------------------+-----------------+-----------------------+----------------+------------------+------------------+%n");  System.out.format(  "|ID | Email | Username | FullName | Department | Position | Create Date |%n");  System.out.format(  "+----+-----------------------+-----------------+-----------------------+----------------+------------------+------------------+%n");  **for** (Account acc : listAcc) {  System.out.format(leftAlignFormat, acc.getId(), acc.getEmail(), acc.getUsername(), acc.getFullName(),  acc.getDepartment().getName(), acc.getPosition().getName(), acc.getCreateDate());  }  System.out.format(  "+----+-----------------------+-----------------+-----------------------+----------------+------------------+------------------+%n");  }  } |

****

# **Tạo @interface CheckDepartmentExists trong package validation:**

|  |
| --- |
| **package** com.vti.validation;  **import** **static** java.lang.annotation.ElementType.ANNOTATION\_TYPE;  **import** **static** java.lang.annotation.ElementType.CONSTRUCTOR;  **import** **static** java.lang.annotation.ElementType.FIELD;  **import** **static** java.lang.annotation.ElementType.LOCAL\_VARIABLE;  **import** **static** java.lang.annotation.ElementType.METHOD;  **import** **static** java.lang.annotation.ElementType.PACKAGE;  **import** **static** java.lang.annotation.ElementType.PARAMETER;  **import** **static** java.lang.annotation.ElementType.TYPE;  **import** **static** java.lang.annotation.ElementType.TYPE\_PARAMETER;  **import** **static** java.lang.annotation.ElementType.TYPE\_USE;  **import** **static** java.lang.annotation.RetentionPolicy.RUNTIME;  **import** java.lang.annotation.Documented;  **import** java.lang.annotation.Retention;  **import** java.lang.annotation.Target;  **import** javax.validation.Constraint;  **import** javax.validation.Payload;  @Target({ METHOD, FIELD, ANNOTATION\_TYPE, CONSTRUCTOR, PARAMETER, TYPE\_USE })  @Retention(RUNTIME)  @Documented  @Constraint(validatedBy = { CheckDepartmentExistsValidator.**class** })  //@Repeatable(List.class)  **public** **@interface** CheckDepartmentExists {  String message() **default** "Department Không có trên hệ thống!!!";  Class<?>[] groups() **default** {};  Class<? **extends** Payload>[] payload() **default** {};  @Target({ METHOD, FIELD, ANNOTATION\_TYPE, CONSTRUCTOR, PARAMETER, TYPE\_USE })  @Retention(RUNTIME)  @Documented  **@interface** List {  CheckDepartmentExists[] value();  }  } |

****

# **Tạo Class CheckDepartmentExistsValidator trong package validation:**

|  |
| --- |
| **package** com.vti.validation;  **import** javax.validation.ConstraintValidator;  **import** javax.validation.ConstraintValidatorContext;  **import** com.vti.datalayer.DepartmentRepository;  **public** **class** CheckDepartmentExistsValidator **implements** ConstraintValidator<CheckDepartmentExists, Integer> {  **private** DepartmentRepository depRepository;  @Override  **public** **void** initialize(CheckDepartmentExists constraintAnnotation) {  depRepository = **new** DepartmentRepository();  }  @Override  **public** **boolean** isValid(Integer depID, ConstraintValidatorContext context) {  Boolean flag = depRepository.isDepartmentExistsByID((**short**) depID.intValue());  **return** flag;  }  } |

****

# **Tạo @interface CheckEmailNotExists trong package validation:**

|  |
| --- |
| **package** com.vti.validation;  **import** **static** java.lang.annotation.ElementType.ANNOTATION\_TYPE;  **import** **static** java.lang.annotation.ElementType.CONSTRUCTOR;  **import** **static** java.lang.annotation.ElementType.FIELD;  **import** **static** java.lang.annotation.ElementType.LOCAL\_VARIABLE;  **import** **static** java.lang.annotation.ElementType.METHOD;  **import** **static** java.lang.annotation.ElementType.PACKAGE;  **import** **static** java.lang.annotation.ElementType.PARAMETER;  **import** **static** java.lang.annotation.ElementType.TYPE;  **import** **static** java.lang.annotation.ElementType.TYPE\_PARAMETER;  **import** **static** java.lang.annotation.ElementType.TYPE\_USE;  **import** **static** java.lang.annotation.RetentionPolicy.RUNTIME;  **import** java.lang.annotation.Documented;  **import** java.lang.annotation.Retention;  **import** java.lang.annotation.Target;  **import** javax.validation.Constraint;  **import** javax.validation.Payload;  @Target({ METHOD, FIELD, ANNOTATION\_TYPE, CONSTRUCTOR, PARAMETER, TYPE\_USE })  @Retention(RUNTIME)  @Documented  @Constraint(validatedBy = { CheckEmailExistsValidator.**class** })  //@Repeatable(List.class)  **public** **@interface** CheckEmailNotExists {  String message() **default** "Email này đã được sử dụng trên hệ thống";  Class<?>[] groups() default {};  Class<? extends Payload>[] payload() default {};  @Target({ METHOD, FIELD, ANNOTATION\_TYPE, CONSTRUCTOR, PARAMETER, TYPE\_USE })  @Retention(RUNTIME)  @Documented  @interface List {  CheckEmailNotExists[] value();  }  } |

****

# **Tạo Class CheckEmailExistsValidator trong package validation:**

|  |
| --- |
| **package** com.vti.validation;  **import** javax.validation.ConstraintValidator;  **import** javax.validation.ConstraintValidatorContext;  **import** com.mysql.cj.util.StringUtils;  **import** com.vti.datalayer.AccountRepository;  **public** **class** CheckEmailExistsValidator **implements** ConstraintValidator<CheckEmailNotExists, String> {  **private** AccountRepository accRepository;  @Override  **public** **void** initialize(CheckEmailNotExists constraintAnnotation) {  accRepository = **new** AccountRepository();  }  @Override  **public** **boolean** isValid(String email, ConstraintValidatorContext context) {  **if** (StringUtils.isNullOrEmpty(email)) {  **return** **false**;  }  Boolean flag = accRepository.isAccountExistsByEmail(email);  **return** !flag;  }  } |

****

