Lab_10

Q.1. Develop a basic Create, read operation using Hibernate for a simple entity, such as Student.

User.java

```
package com.sms;
import javax.persistence.*;
import java.util.Date;
@Entity
@Table(name = "users")
public class User {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long userId;
    @Column(nullable = false, unique = true)
    private String username;
    @Column(nullable = false)
    private String password;
    @Column(nullable = false, unique = true)
    private String email;
    @Temporal(TemporalType.TIMESTAMP)
    private Date dateJoined;

    // Getters and Setters

    public Long getUserId() {
        return userId;
    }

    public void setUserId(Long userId) {
        this.userId = userId;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

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

    public String getEmail() {
        return email;
    }

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

    public Date getDateJoined() {
        return dateJoined;
    }

    public void setDateJoined(Date dateJoined) {
        this.dateJoined = dateJoined;
    }
}
```

App.java

```
package com.sms;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;

import java.util.Calendar;

public class App {
    public static void main(String[] args) {
        // Get the SessionFactory
        SessionFactory sessionFactory = HibernateUtil.getSessionFactory();

        // Perform user operations
        performUserOperations(sessionFactory);
    }
    private static void performUserOperations(SessionFactory sessionFactory) {
        Session session = null;
        Transaction transaction = null;
        try {
            // Open a new session
            session = sessionFactory.openSession();
            // Start a transaction
            transaction = session.beginTransaction();

            // Create a new User object
            User user = new User();
            user.setEmail("poorva@gmail.com");
            user.setUsername("Poorvaja");
            user.setPassword("1234");

            // Set a specific date for dateJoined
            Calendar calendar = Calendar.getInstance();
            calendar.set(2024, Calendar.JULY, 27); // Setting date to July 27,
2024
            user.setDateJoined(calendar.getTime());

            // Save the user to the database
            session.save(user);

            // Commit the transaction
            transaction.commit();

            System.out.println("User saved successfully with ID: " +
user.getUserId());
        } catch (Exception e) {
            if (transaction != null) {
                transaction.rollback();
            }
            e.printStackTrace();
        } finally {
            if (session != null) {
                session.close();
            }
        }
    }
}
```

```
mysql> select * from users;
+--------+----------------------------+------------------+----------+----------+
| userId | dateJoined                 | email            | password | username |
+--------+----------------------------+------------------+----------+----------+
|      5 | 2024-07-27 00:51:44.801000 | Poorva@gmail.com | 1234     | Poorvaja |
+--------+----------------------------+------------------+----------+----------+
1 row in set (0.00 sec)
```

```
Hibernate:

    create table users (
       userId bigint not null auto_increment,
        dateJoined datetime(6),
        email varchar(255) not null,
        password varchar(255) not null,
        username varchar(255) not null,
        primary key (userId)
    ) engine=InnoDB
```

```
INFO: HHH10001501: Connection obtained from JdbcConnect
Hibernate:
    insert
    into
        users
        (dateJoined, email, password, username)
    values
        (?, ?, ?, ?)
```

Lab_10

Q.2. Use get() method to fetch a student object with an ID that doesn't exist in the database. What will be the result, and how would you handle it?

User.java

```
package com.sms;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;

import java.util.Calendar;

public class App {
    public static void main(String[] args) {
        // Get the SessionFactory
        SessionFactory sessionFactory = HibernateUtil.getSessionFactory();

        // Perform user operations
        performUserOperations(sessionFactory);
    }

    private static void performUserOperations(SessionFactory sessionFactory) {
        Session session = null;
        Transaction transaction = null;
        try {
            // Open a new session
            session = sessionFactory.openSession();
            // Start a transaction
            transaction = session.beginTransaction();

            // Create a new User object
```

```
        User user = new User();
        user.setEmail("poorva@gmail.com");
        user.setUsername("Poorvaja");
        user.setPassword("1234");

        // Set a specific date for dateJoined
        Calendar calendar = Calendar.getInstance();
        calendar.set(2024, Calendar.JULY, 27); // Setting date to July 27, 2024
        user.setDateJoined(calendar.getTime());

        // Save the user to the database
        session.save(user);

        // Commit the transaction
        transaction.commit();

        System.out.println("User saved successfully with ID: " + user.getUserId());

        // Fetch a user by ID that doesn't exist
        Long nonExistentUserId = 999L; // Assume this ID doesn't exist
        User fetchedUser = getUserById(session, nonExistentUserId);

        // Handle the case where the user is not found
        if (fetchedUser == null) {
            System.out.println("User with ID " + nonExistentUserId + " not found.");
        } else {
            System.out.println("Fetched User: " + fetchedUser.getUsername());
        }
    } catch (Exception e) {
        if (transaction != null) {
            transaction.rollback();
```

```
        }
        e.printStackTrace();
      } finally {
        if (session != null) {
          session.close();
        }
      }
    }

    // Method to fetch a User by ID
    public static User getUserById(Session session, Long userId) {
      return session.get(User.class, userId);
    }
}
```

```
mysql> select * from users;
+--------+----------+----------+------------------+---------------------+
| userId | username | password | email            | dateJoined          |
+--------+----------+----------+------------------+---------------------+
|      5 | Poorvaja | 1234     | poorva@gmail.com | 2024-07-27 23:10:29 |
+--------+----------+----------+------------------+---------------------+
1 row in set (0.00 sec)
```

If the parameter (ID=”999”) of Object is not match with database’s record then eclipse shows : user not found.

```
Aug 03, 2024 11:10:29 PM org.hibernate.resource.transactic
INFO: HHH10001501: Connection obtained from JdbcConnectior
Hibernate:
    insert
    into
        users
        (dateJoined, email, password, username)
    values
        (?, ?, ?, ?)
User saved successfully with ID: 5
Hibernate:
    select
        user0_.userId as userid1_0_0_,
        user0_.dateJoined as datejoin2_0_0_,
        user0_.email as email3_0_0_,
        user0_.password as password4_0_0_,
        user0_.username as username5_0_0_
    from
        users user0_
    where
        user0_.userId=?
User with ID 999 not found.
```

```
mysql> select * from users;
+--------+----------+----------+------------------+---------------------+
| userId | username | password | email            | dateJoined          |
+--------+----------+----------+------------------+---------------------+
|      5 | Poorvaja | 1234     | poorva@gmail.com | 2024-07-27 23:10:29 |
|      7 | ABCD     | 1234     | abcd@gmail.com   | 2024-07-27 23:28:16 |
+--------+----------+----------+------------------+---------------------+
```

If the parameter (ID=”5”) of Object is not match with database’s record then eclipse shows : username.

```
Aug 03, 2024 11:28:15 PM org.hibernate.resource.trans
INFO: HHH10001501: Connection obtained from JdbcConne
Hibernate:
    insert
    into
        users
        (dateJoined, email, password, username)
    values
        (?, ?, ?, ?)
User saved successfully with ID: 7
Hibernate:
    select
        user0_.userId as userid1_0_0_,
        user0_.dateJoined as datejoin2_0_0_,
        user0_.email as email3_0_0_,
        user0_.password as password4_0_0_,
        user0_.username as username5_0_0_
    from
        users user0_
    where
        user0_.userId=?
Fetched User: Poorvaja
```

Q.3. Also demonstrate use of load() method.

```
package com.sms;

import org.hibernate.Session;
import org.hibernate.ObjectNotFoundException;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;

import java.util.Calendar;

public class App {
    public static void main(String[] args) {
        // Get the SessionFactory
        SessionFactory sessionFactory = HibernateUtil.getSessionFactory();

        // Perform user operations
        performUserOperations(sessionFactory);
    }

    private static void performUserOperations(SessionFactory sessionFactory) {
        Session session = null;
        Transaction transaction = null;
        try {
            // Open a new session
            session = sessionFactory.openSession();
            // Start a transaction
            transaction = session.beginTransaction();

            // Create a new User object
            User user = new User();
            user.setEmail("EFG@gmail.com");
            user.setUsername("EFG");
            user.setPassword("1234");

            // Set a specific date for dateJoined
            Calendar calendar = Calendar.getInstance();
            calendar.set(2024, Calendar.JULY, 27); // Setting date to July 27,
2024
            user.setDateJoined(calendar.getTime());

            // Save the user to the database
            session.save(user);

            // Commit the transaction
            transaction.commit();

//            System.out.println("User saved successfully with ID: " +
user.getUserId());

            // Fetch a user by ID that doesn't exist
//            Long nonExistentUserId = 5L; // Assume this ID exist
//            User fetchedUser = getUserById(session, nonExistentUserId);

            // Handle the case where the user is not found
//            if (fetchedUser == null) {
```

```
//                System.out.println("User with ID " + nonExistentUserId + " not
found.");
//            } else {
//                System.out.println("Fetched User: " +
fetchedUser.getUsername());
//            }
            demonstrateLoadMethod(session,18L);
        } catch (Exception e) {
            if (transaction != null) {
                transaction.rollback();
            }
            e.printStackTrace();
        } finally {
            if (session != null) {
                session.close();
            }
        }
    }

    // Method to fetch a User by ID
    public static User getUserById(Session session, Long userId) {
        return session.get(User.class, userId);
    }

 // Method to demonstrate the use of load() method
    public static void demonstrateLoadMethod(Session session, Long userId) {
        try {
            // Fetch the user using load() method
            User loadedUser = session.load(User.class, userId);
            System.out.println("User loaded successfully with ID: " +
loadedUser.getUserId());

            // Accessing a property other than the identifier to initialize the
proxy
            System.out.println("Username: " + loadedUser.getUsername());
            System.out.println(":email" + loadedUser.getEmail());
        } catch (ObjectNotFoundException e) {
            System.out.println("User with ID " + userId + " not found. Exception:
" + e.getMessage());
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            if (session != null) {
                session.close();
            }
        }
    }
}
```

```
+--------+----------+----------+------------------+---------------------+
| userId | username | password | email            | dateJoined          |
+--------+----------+----------+------------------+---------------------+
|      5 | Poorvaja | 1234     | poorva@gmail.com | 2024-07-27 23:10:29 |
+--------+----------+----------+------------------+---------------------+
1 row in set (0.00 sec)
```

If the parameter (userID=”18L”) of Object is not match with database’s record then eclipse ran the program successfully and shows an exception:

```
User loaded successfully with ID: 18
Hibernate:
    select
        user0_.userId as userid1_0_0_,
        user0_.dateJoined as datejoin2_0_0_,
        user0_.email as email3_0_0_,
        user0_.password as password4_0_0_,
        user0_.username as username5_0_0_
    from
        users user0_
    where
        user0_.userId=?
User with ID 18 not found. Exception: No row with the given identifier exists: [com.sms.User#18]
```

If the parameter (userID=”5L”) of Object is match with database’s record then eclipse ran the program successfully and shows the record:

```
User loaded successfully with ID: 5
Hibernate:
    select
        user0_.userId as userid1_0_0_,
        user0_.dateJoined as datejoin2_0_0_,
        user0_.email as email3_0_0_,
        user0_.password as password4_0_0_,
        user0_.username as username5_0_0_
    from
        users user0_
    where
        user0_.userId=?
Username: Poorvaja
:emailpoorva@gmail.com
```