

PROGRAM LANJUT

Java Persistence API

Oleh Politeknik Elektronika Negeri Surabaya

2017



**Politeknik Elektronika Negeri Surabaya
Departemen Teknik Informatika dan Komputer**

Konten

- Pengenalan Java Persistence API
- Object Model vs Relational Model
- Apa itu JPA
- Object Relational Mapping
- Percobaan

Pengenalan Java Persistence API

- Setiap aplikasi enterprise melakukan operasi database dengan menyimpan dan mengambil data yang cukup besar.
- Meskipun telah banyak tersedia teknologi untuk menajemen penyimpanan data, namun para pengembang aplikasi tetap harus bersusah payah untuk membuat operasi database yang efisien.
- Terlebih lagi saat ini kebanyakan aplikasi telah berbasis obyek. Dimana data yang digunakan dalam database berbentuk obyek. Sedangkan penyimpanan dalam database menggunakan bentuk relational data.
- Tentunya hal ini akan membuat pekerjaan pengembang aplikasi semakin rumit.



Pengenalan Java Persistence API

- Java Persistence API (JPA) membantu pengembang aplikasi untuk memanajemen database.
- Dengan menggunakan JPA, interaksi dengan database berkurang secara signifikan.
- JPA juga menjadi perantara antara object model dan relational model.



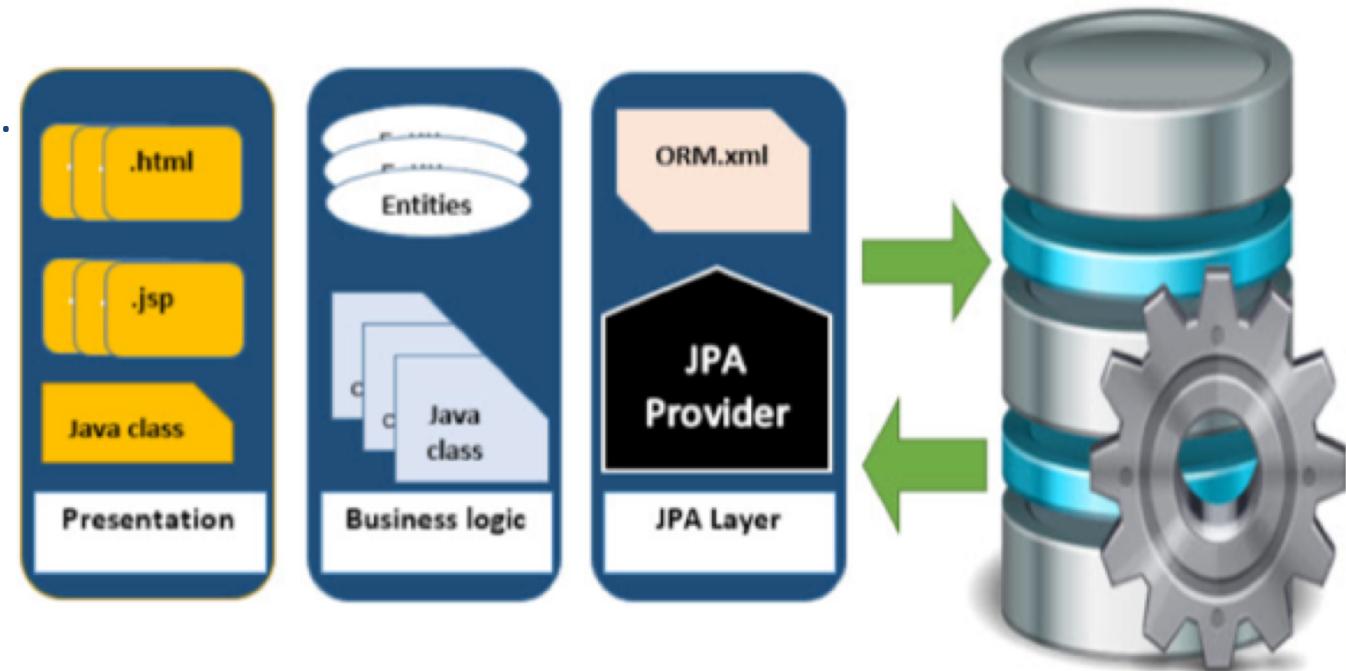
Object Model vs Relational Model

- Relational model diwakili dalam format tabular, sedangkan object model diwakili dalam format object yang saling terhubung.
- Saat menyimpan dan mengambil model object dari relational database, beberapa ketidaksesuaian terjadi. Penyebabnya adalah:
 - Granularity
Model object memiliki granularity yang lebih dibandingkan dengan relational model.
 - Subtypes (Inheritance)
Subtipe atau pewarisan tidak didukung oleh semua jenis database relasional
 - Associations
Sebuah object dapat berasosiasi dengan object lain, di dalam sebuah object memungkinkan terdapat beberapa object lain. Sedangkan dalam relational model tidak memungkinkan terdapat tabel di dalam tabel. Relational tabel hanya dapat menerima foreign key sebagai relasi.
 - Data navigation
Object model dan relational model memiliki navigasi data yang berbeda



Apa itu JPA

- JPA (Java Persistence API) adalah sekumpulan class dan method untuk menyimpan data kedalam database
- JPA dibuat untuk mengurangi beban penulisan kode program untuk pengelolaan database, yang memungkinkan interaksi yang mudah dengan database.
- Gambar berikut menjelaskan letak dari JPA didalam sebuah sistem.



Object Relational Maping (ORM)

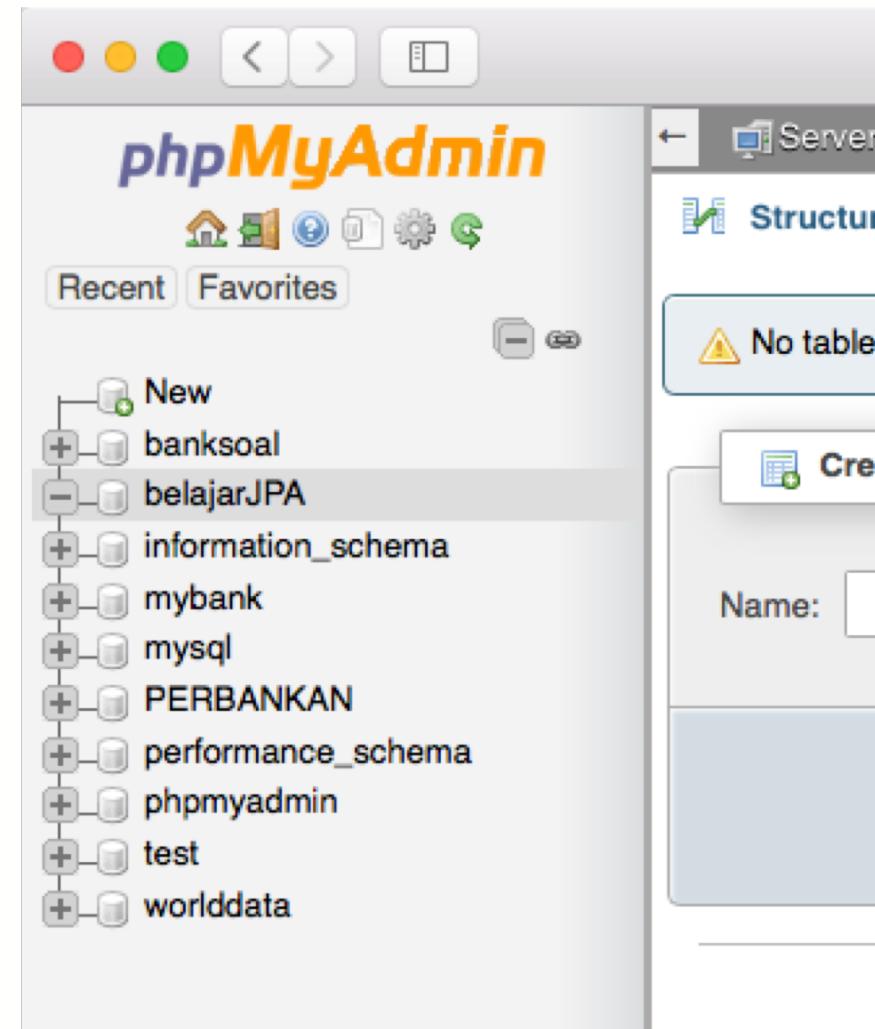
- ORM merupakan salah satu komponen penting di dalam JPA.
- ORM merupakan sebuah program untuk merubah data object menjadi data relational dan sebaliknya.
- Dengan adanya ORM kita dapat menggunakan data Object (Model/Entity/POJO/Bean) di dalam aplikasi kita dan tetap menggunakan relational database untuk penyimpananya, tanpa perlu khawatir untuk melakukan konversi dari object model menjadi relational.



PERCOBAAN

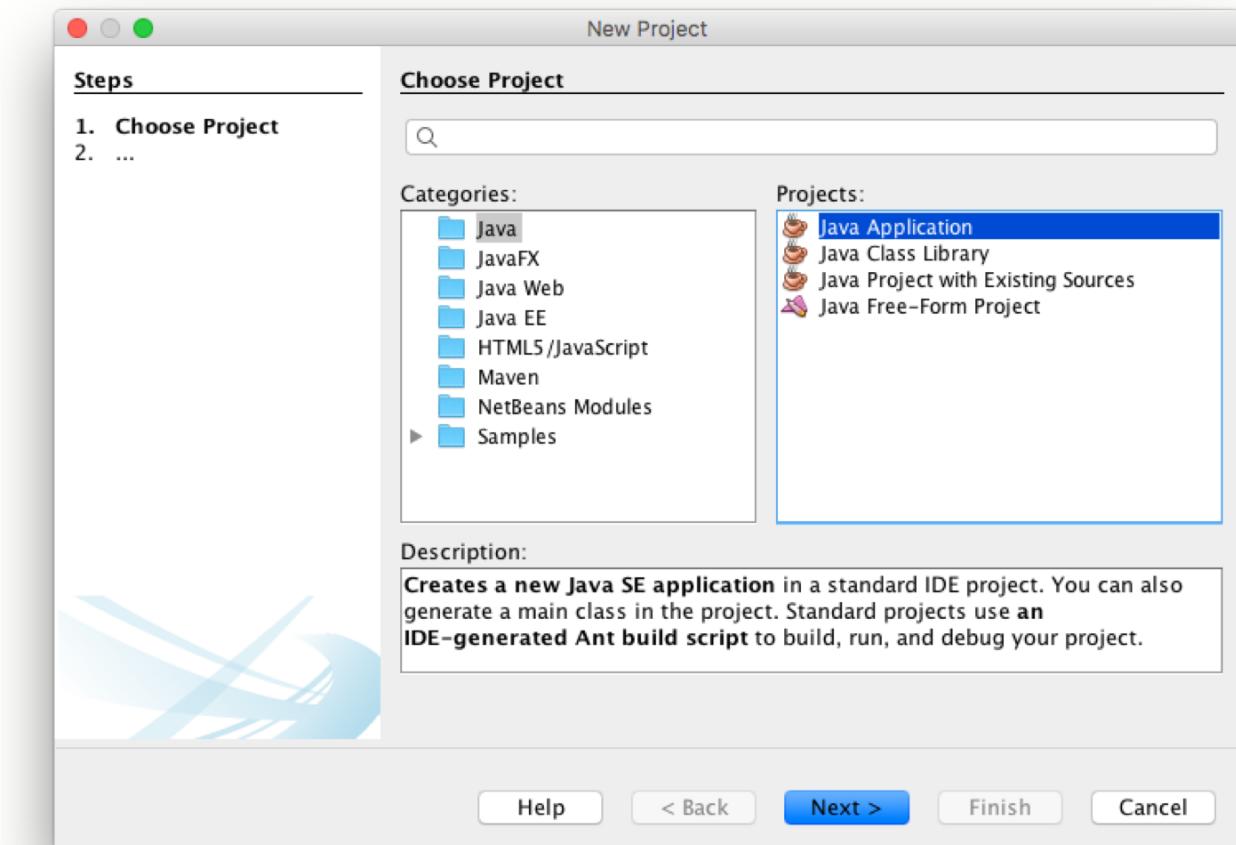
Percobaan JPA

Buatlah sebuah database baru dengan nama belajarJPA



Percobaan JPA

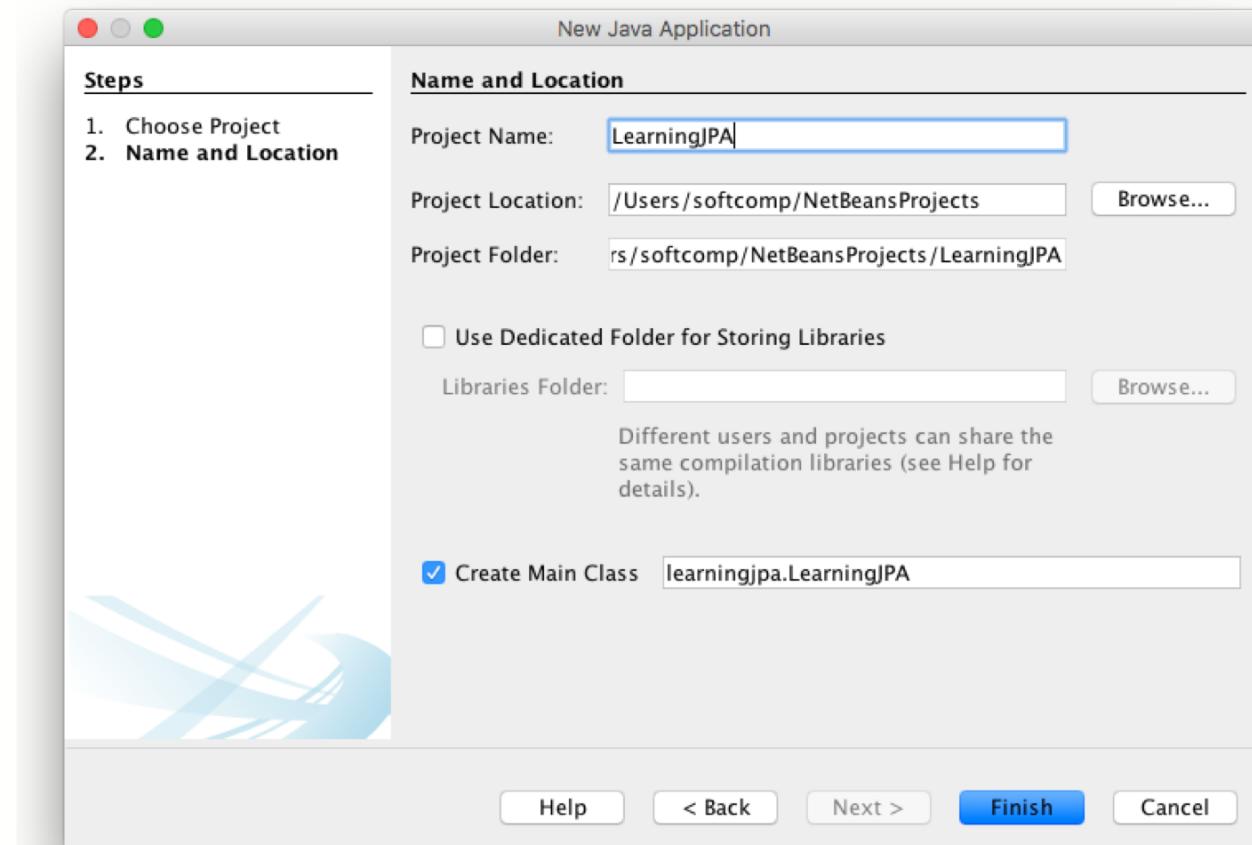
Buatlah project java baru pada netbean



Percobaan JPA

Namai project anda dengan nama
LearningJPA

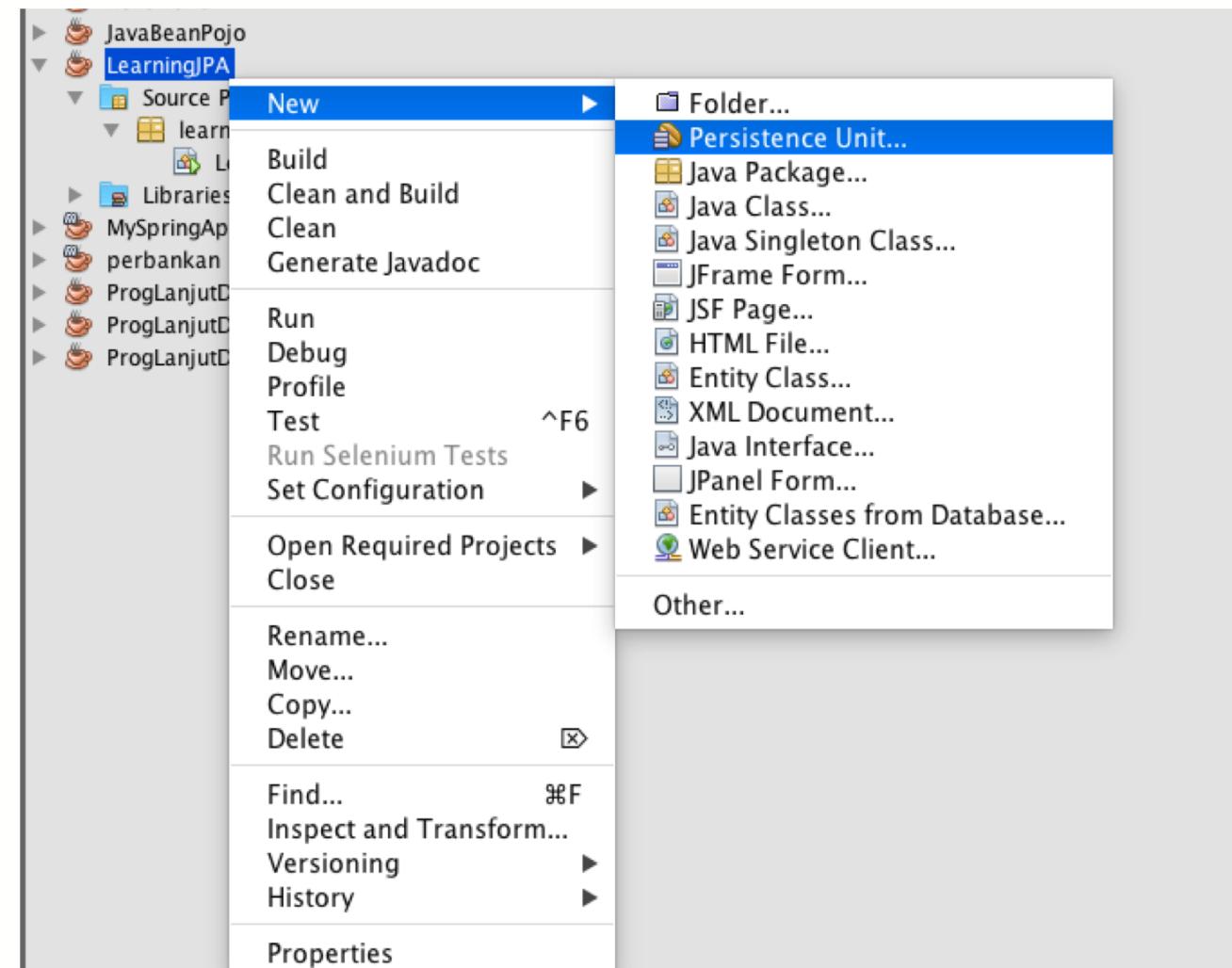
Kemudian klik Finish



Percobaan JPA

Klik kanan pada project baru anda kemudian pilihlah New > Persistance Unit...

(Jika pilahan Persistance Unit tidak muncul, carilah pilihan tersebut pada menu Others...)



Percobaan JPA

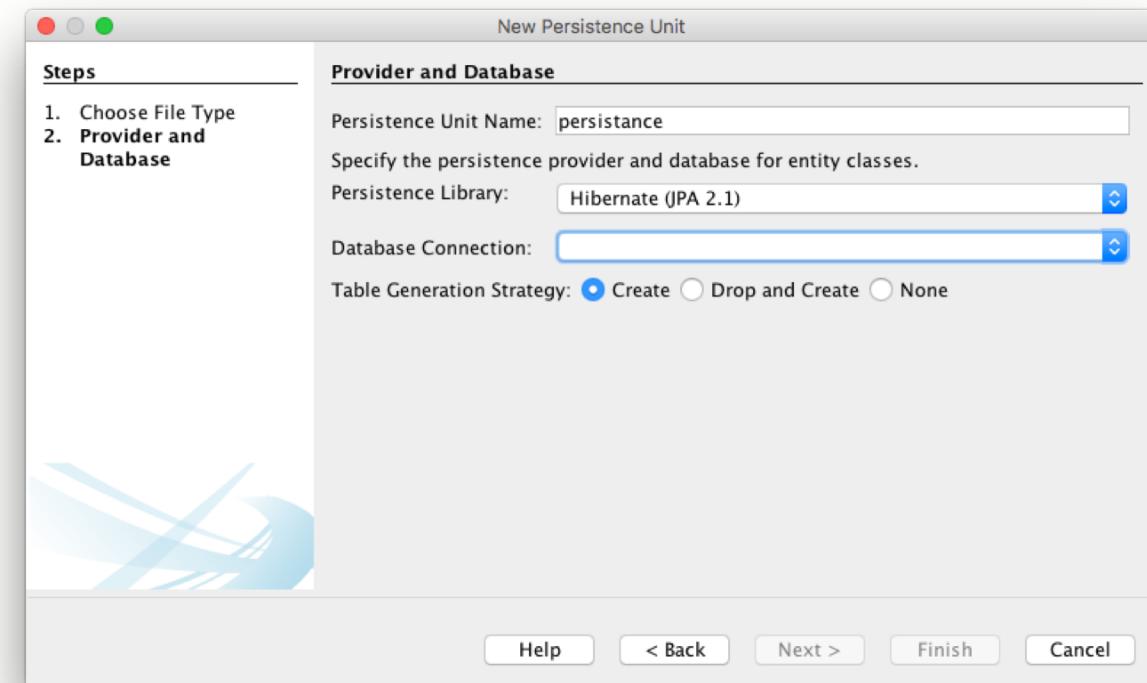
Namai persistance unit anda dengan nama

Persistance Unit Name: **persistence**

Kemudian pilihlah hibernate sebagai library

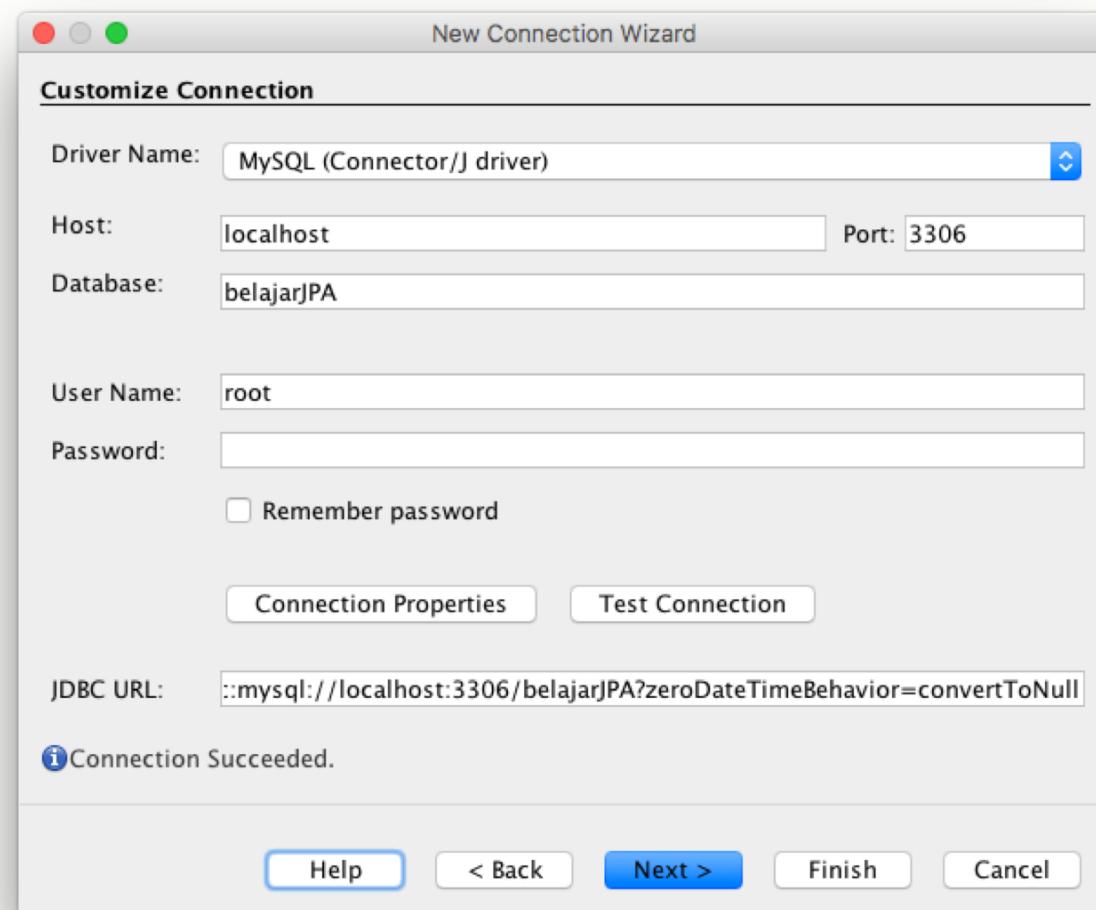
Persistance Library: **EclipseLink (JPA 2.1)**

Kemudian untuk Database Connection
pilihlah **New Database Connection**



Percobaan JPA

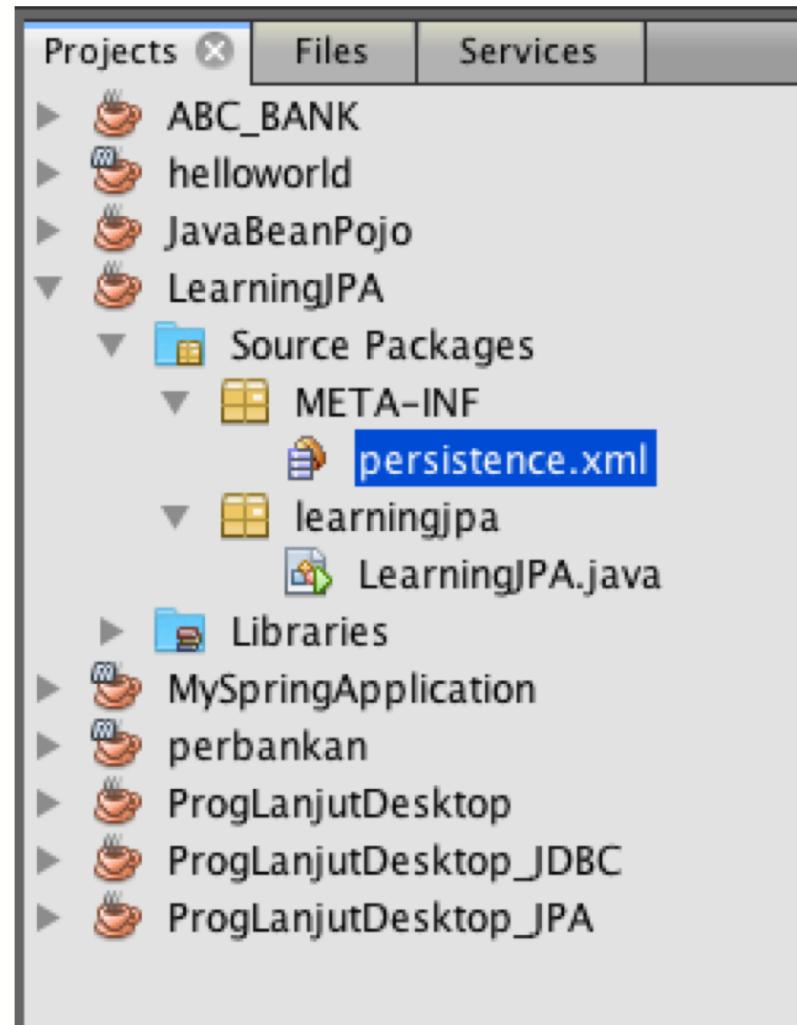
Ubahlah nama database menjadi
Database: **belajarJPA**
Kemudian klik **Next > Next > Finish > Finish**



Percobaan JPA

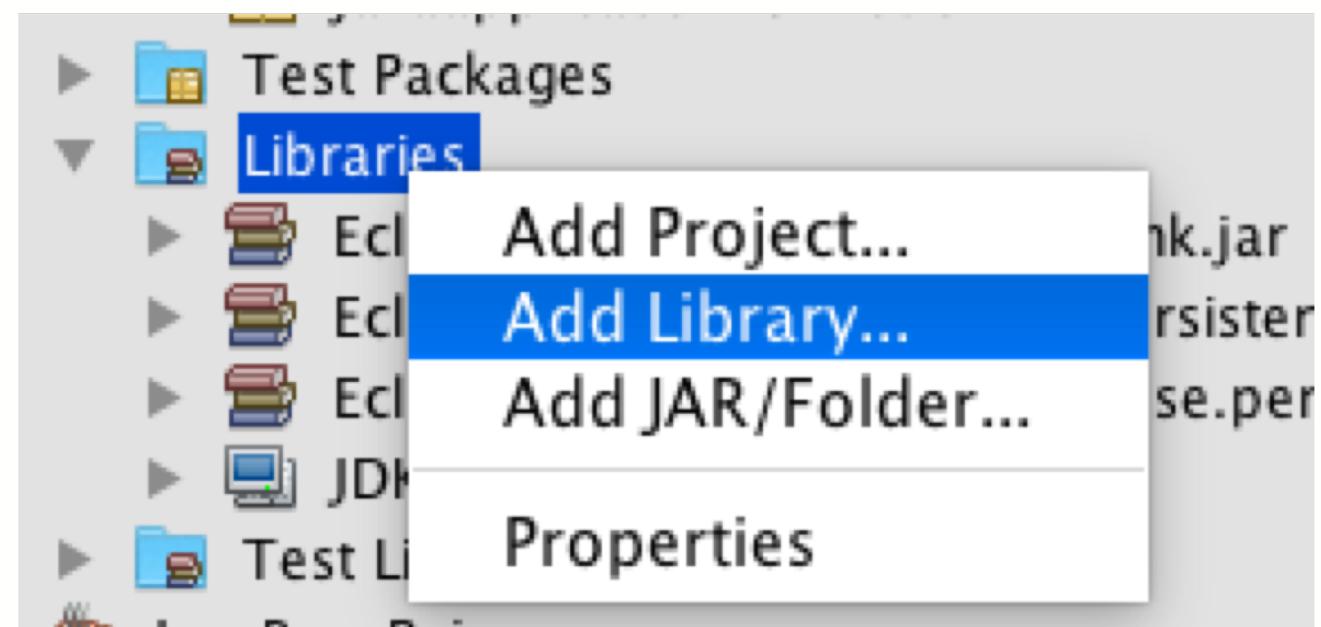
Akan muncul folder baru pada project anda
“META-INF”

Dimana folder tersebut berisikan file
“persistance.xml”



Percobaan JPA

Klik kanan pada folder Libraries, kemudian pilihlah **Add Library...**



Percobaan JPA

Tambahkan library MySQL JDBC Driver

Available Libraries:

- JSFL 1.2.1
- JUnit 4.12
- JWS Ant Tasks
- METRO 2.0
- MySQL JDBC Driver**
- Persistence (JPA 2.1)
- Persistence JPA2.1
- PostgreSQL JDBC Driver
- PrimeFaces 5.0
- Spring Framework 3.2.7
- Spring Framework 4.0.1
- Spring Web MVC 3.2.7
- Spring Web MVC 4.0.1
- Struts 1.3.10
- Swing Layout Extensions
- TestNG 6.8.1

Create...

Add Library

Cancel

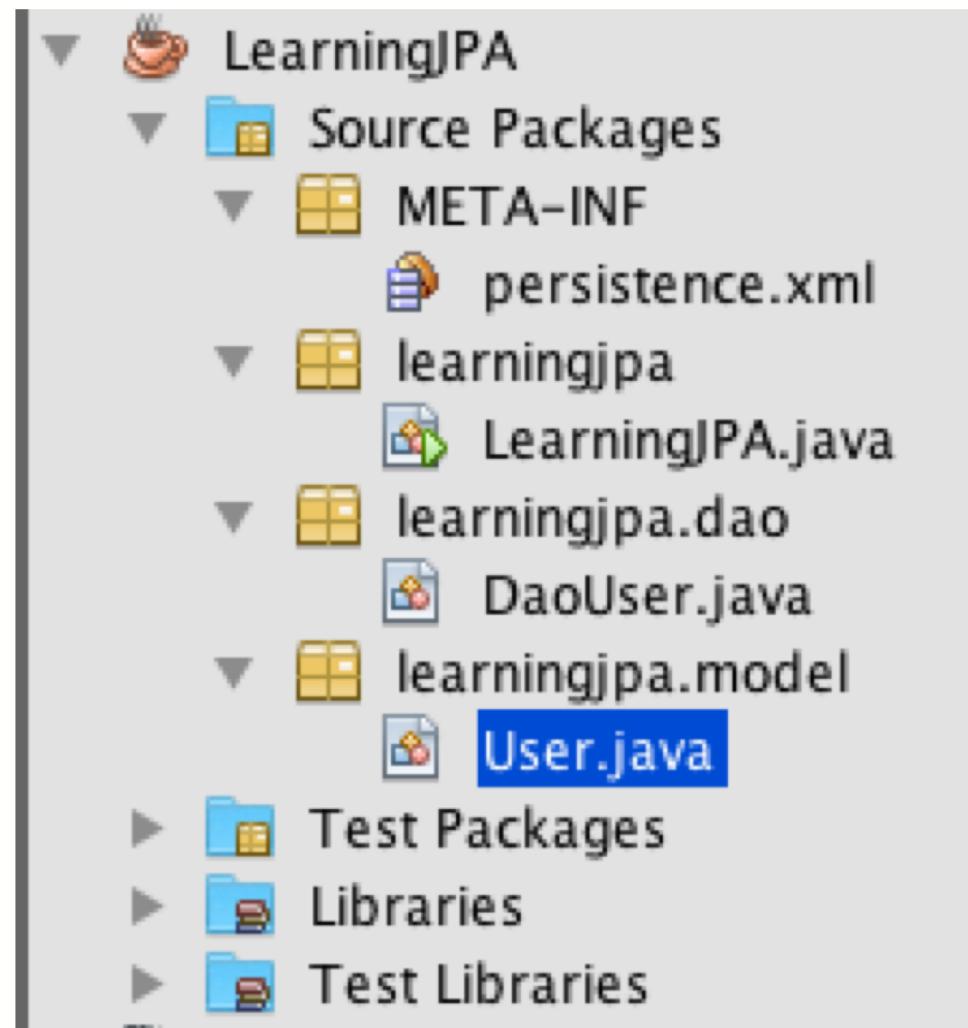


Percobaan JPA

Kemudian buatlah beberapa folder dan file dalam aplikasi anda seperti pada gambar

Anda perlu menambahkan folder **dao** dan folder **model**

Anda juga perlu menambahkan file **User.java** dan **DaoUser.java**



Percobaan JPA

Tuliskan kode berikut pada file **User.java**

```
package learningjpa.model;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

@Entity
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;

    private String nama;

    public User() {}

    public String getNama() {
        return nama;
    }

    public void setNama(String nama) {
        this.nama = nama;
    }

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }
}
```



Percobaan JPA

Tuliskan kode berikut pada file **DaoUser.java**

```
package learningjpa.dao;

import java.util.List;
import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;
import javax.persistence.TypedQuery;
import javax.persistence.criteria.CriteriaBuilder;
import javax.persistence.criteria.CriteriaQuery;
import javax.persistence.criteria.Root;
import learningjpa.model.User;

public class DaoUser {
    public void save(User user) {
        EntityManagerFactory emfactory = Persistence.createEntityManagerFactory("persistance");

        EntityManager entitymanager = emfactory.createEntityManager();
        entitymanager.getTransaction().begin();

        entitymanager.persist(user);
        entitymanager.getTransaction().commit();

        entitymanager.close();
        emfactory.close();
    }

    public void update(long id, User user){
        EntityManagerFactory emfactory = Persistence.createEntityManagerFactory("persistance");

        EntityManager entitymanager = emfactory.createEntityManager();
        entitymanager.getTransaction().begin();
        User editUser = entitymanager.find(User.class, id);

        //edit all field except id
        editUser.setNama(user.getNama());

        entitymanager.getTransaction().commit();

        entitymanager.close();
        emfactory.close();
    }
}
```



Percobaan JPA

```
public void delete(long id){  
    EntityManagerFactory emfactory = Persistence.createEntityManagerFactory("persistance");  
    EntityManager entitymanager = emfactory.createEntityManager();  
    entitymanager.getTransaction().begin();  
  
    User user = entitymanager.find(User.class, id);  
    entitymanager.remove(user);  
    entitymanager.getTransaction().commit();  
    entitymanager.close();  
    emfactory.close();  
}  
  
public User findById(long id){  
  
    EntityManagerFactory emfactory = Persistence.createEntityManagerFactory("persistance");  
    EntityManager entitymanager = emfactory.createEntityManager();  
    User user = entitymanager.find(User.class, id);  
  
    return user;  
}  
public List<User> findAll(){  
  
    EntityManagerFactory emfactory = Persistence.createEntityManagerFactory("persistance");  
    EntityManager entitymanager = emfactory.createEntityManager();  
    CriteriaBuilder cb = entitymanager.getCriteriaBuilder();  
    CriteriaQuery<User> cq = cb.createQuery(User.class);  
    Root<User> rootEntry = cq.from(User.class);  
    CriteriaQuery<User> all = cq.select(rootEntry);  
    TypedQuery<User> allQuery = entitymanager.createQuery(all);  
    return allQuery.getResultList();  
}  
}
```



Percobaan JPA

Tambahkan kode berikut pada file **LearningJPA.java**

```
package learningjpa;

import java.util.List;
import learningjpa.dao.DaoUser;
import learningjpa.model.User;

public class LearningJPA {

    public static void main(String[] args) {
        //membuat object DaoUser untuk data akses
        DaoUser daoUser = new DaoUser();

        //buat user baru dan simpan pada database
        User user = new User();
        user.setNama("Fahrul");
        daoUser.save(user);

        //tampilkan data yang telah terseimpan
        List<User> result = daoUser.findAll();
        for(int i=0; i<result.size();i++){
            User printUser = result.get(i);
            System.out.println(printUser.getId()+"-"+printUser.getNama());
        }
    }
}
```

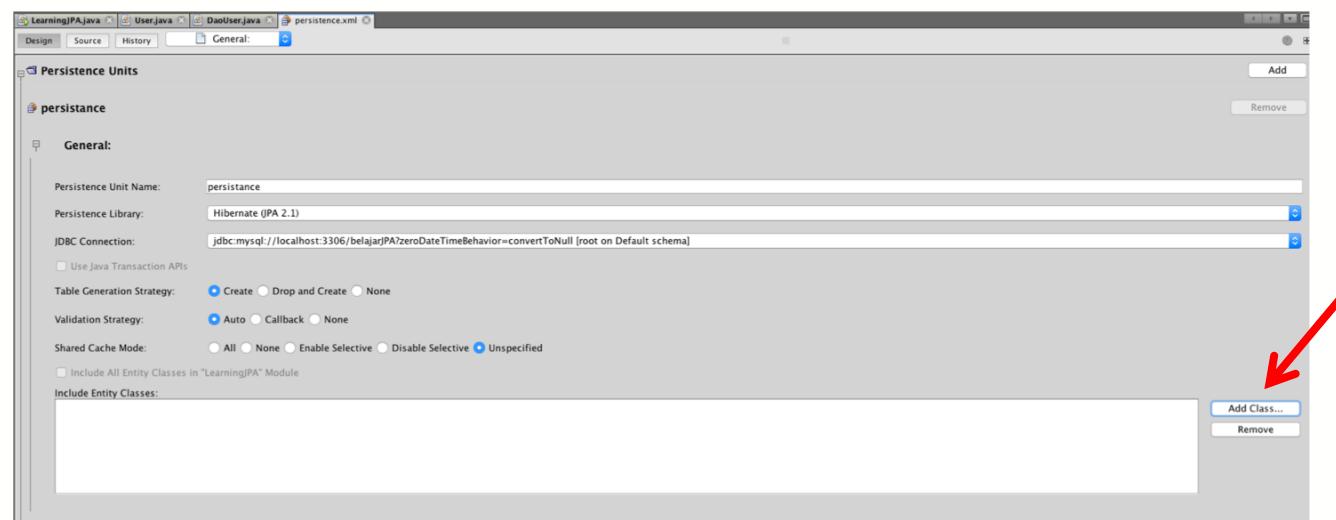


Percobaan JPA

Registrasikan file User.java pada persistance.xml

Bukalah file persistance.xml

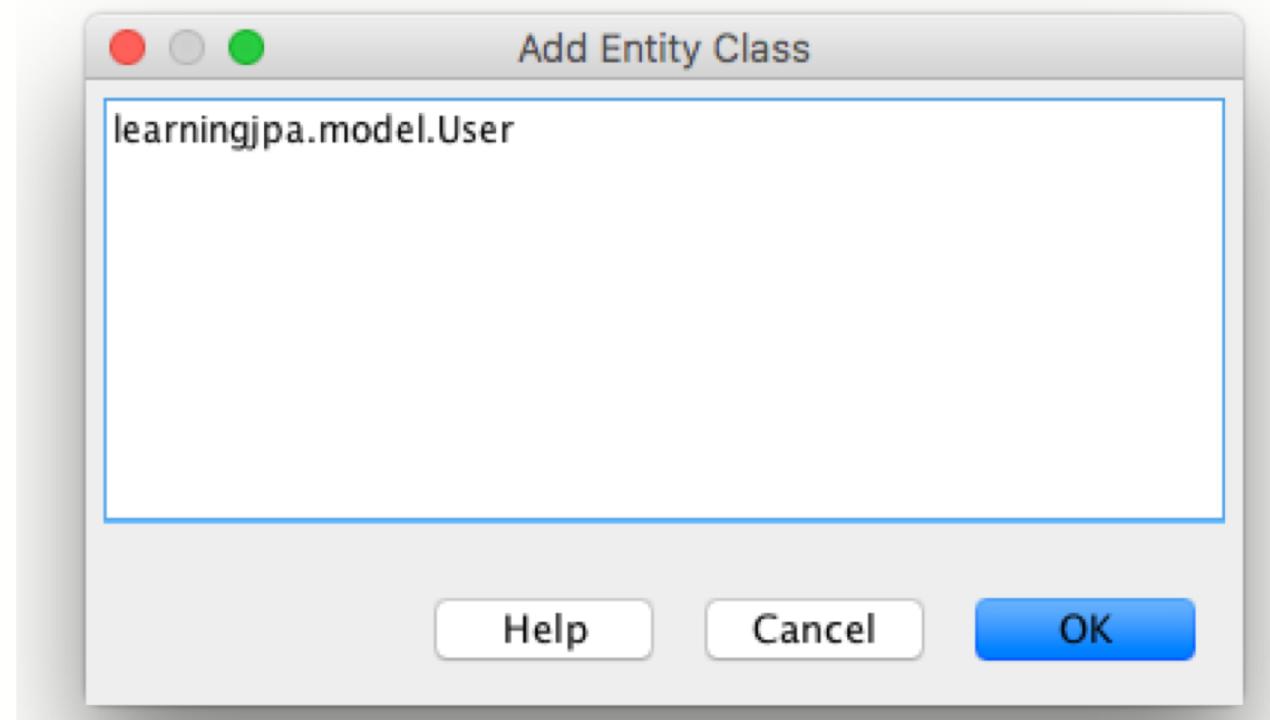
Kemudian pilihlah tombol add class (berada dibagian kanan)



Percobaan JPA

Kemudian pilihlah model yang telah anda buat. Yaitu learningjpa.model.User

Kemudian klik OK



Percobaan JPA

Hasil running program

```
Output - JavaApplication10 (run) ×
run:
[EL Info]: 2017-11-26 19:26:04.099--ServerSession(1904720585)--EclipseL
[EL Info]: connection: 2017-11-26 19:26:04.515--ServerSession(190472058
[EL Info]: connection: 2017-11-26 19:26:04.611--ServerSession(190472058
[EL Info]: 2017-11-26 19:26:04.619--ServerSession(125549276)--EclipseLi
[EL Info]: connection: 2017-11-26 19:26:04.661--ServerSession(125549276
|||||||DATA DALAM DATABASE|||||||
1-Fahrul
51-Fadilah
101-Hardiansyah
|||||||
BUILD SUCCESSFUL (total time: 2 seconds)
```



Referensi

- TutorialsPoint, 2017, “JPA Tutorial”,
<https://www.tutorialspoint.com/jpa/>, (diakses tanggal 26 November 2017)



bridge to the future



<http://www.eepis-its.edu>