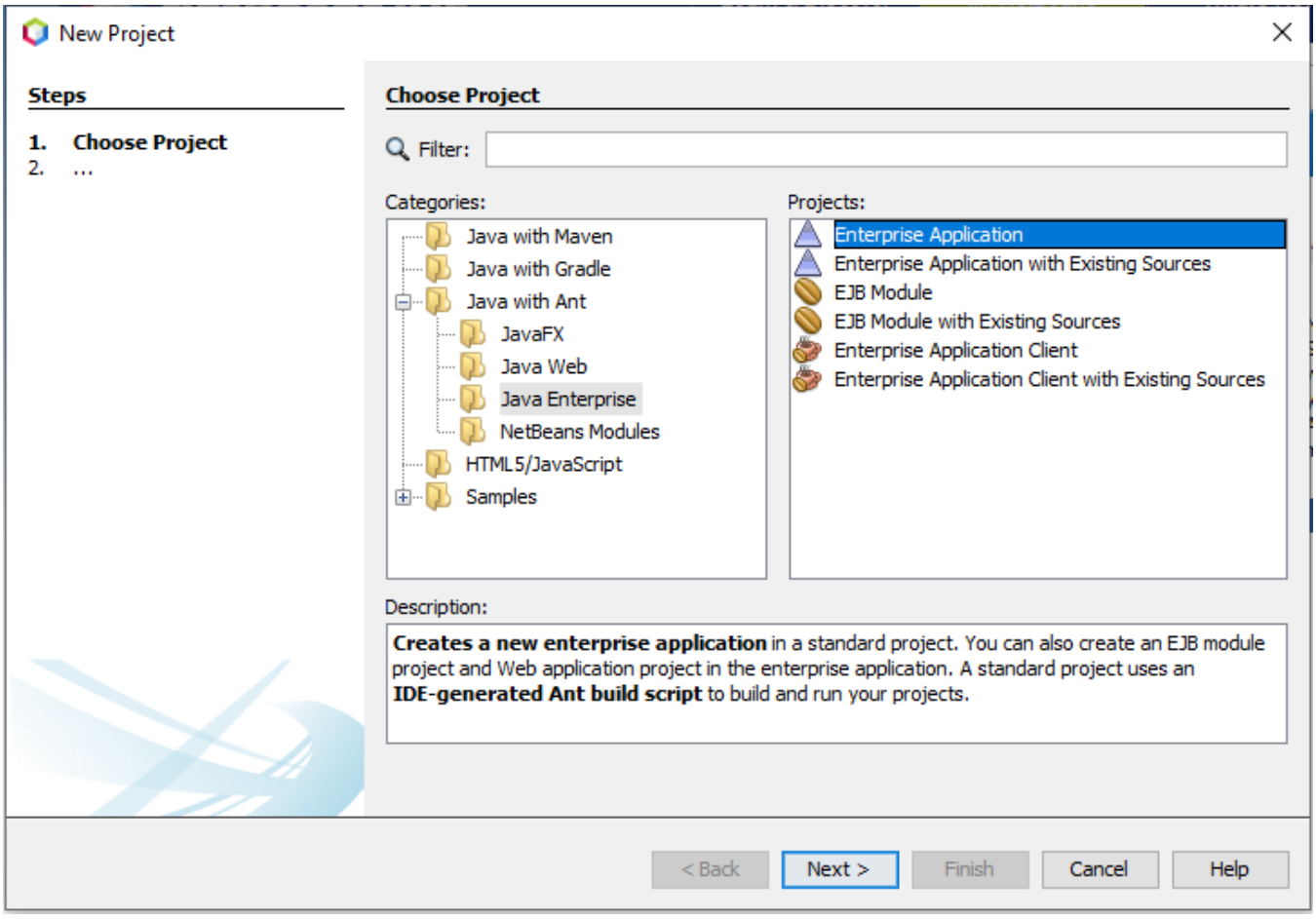
Sau khi thực hành xong bài này, học viên có khả năng :

- *Trình bày được các bước xây dựng project EJB bằng Ant và chạy trên Glassfish 4.x*
- *Tạo ra 1 Session Bean để quản lý danh bạ*
- *Xây dựng 1 Persistance để kết nối đến database.*

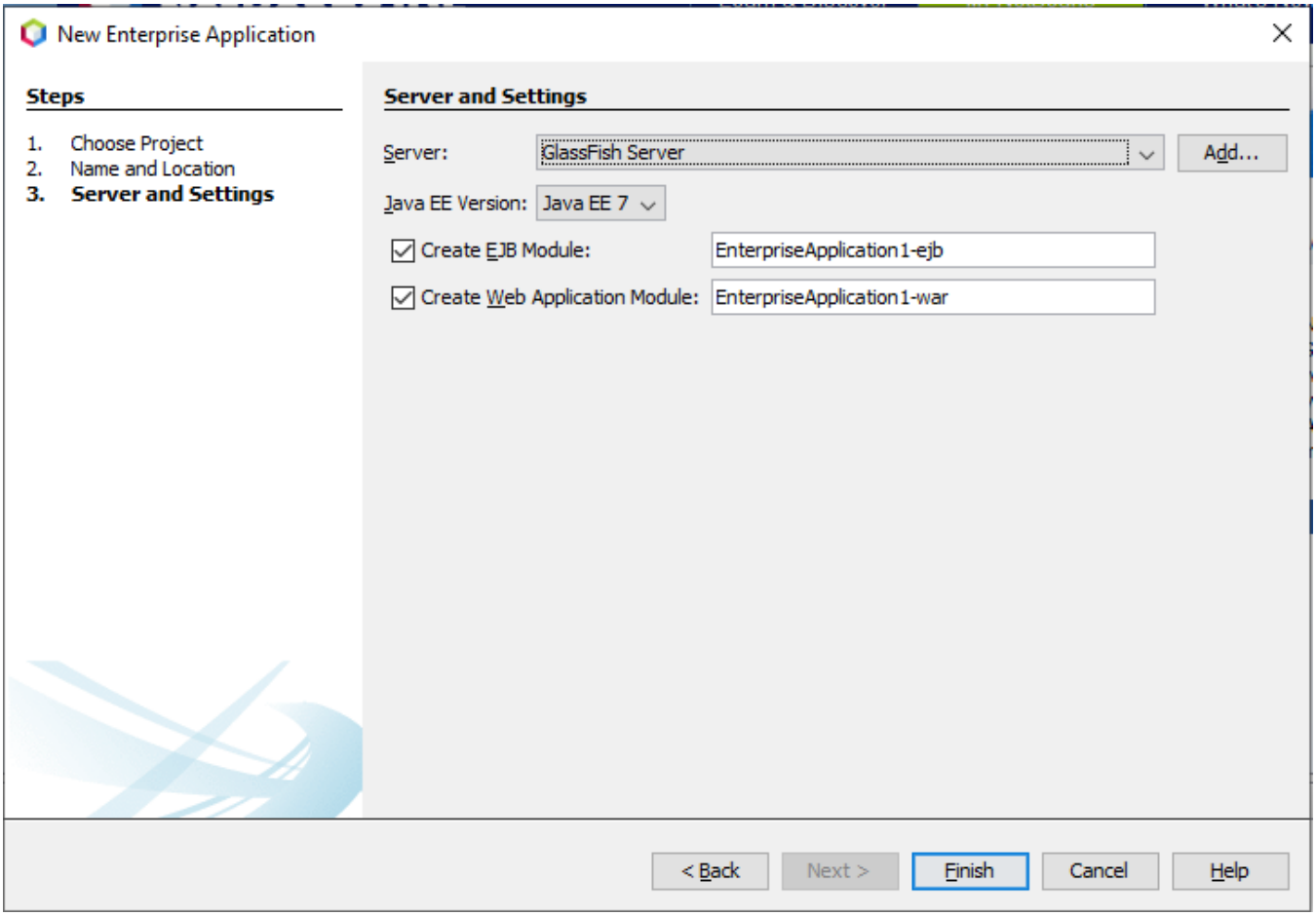
Xây dựng ứng dụng Enterprise Application theo các bước như sau :

Bước 1. Mở ứng dụng Netbean,chọn File | New Project |…| Enterprise Application ,sau đó bấm Next



- Đặt tên cho dự án của mình trong ô Project Name
- Chọn nơi lưu trữ cho dự án của mình trong ô Project Location
- Đặt tên folder trong ô Project Folder

- Sau đó chọn Next
- Tiếp theo ở bước này ta chọn server để chạy , và phiên bản của Java , thường ta sẽ giữ nguyên ở bước này , Chọn Next



- Bấm Finish , và đợt netbean sẽ tự động khởi tạo project cho chúng ta
- Chọn build | run để chạy thử project trên server Glassfish

Bước 2. : Tạo database tên Lab6DB chứa bảng Contact gồm các cột:
(Id: int identity, FirstName: varchar(30), LastName: varchar(30), Email: varchar(30), Phone varchar(20))

Bước 3: Tao ra các Entity

- Contact như sau :

```
@Entity
@Table(name= "contact")
@Cache (usage=CacheConcurrencyStrategy.TRANSACTIONAL)
@EntityListeners({ContactDebugListener.class})
@NamedQueries({
    @NamedQuery(name="Contact.findAll",
                query="SELECT c FROM Contact c"),
    @NamedQuery(name="Contact.findByName",
                query="SELECT c FROM Contact c WHERE c.firstName = :firstName"),
    @NamedQuery(name="Contact.findByEmail",
                query="SELECT c FROM Contact c WHERE c.email = :email")
})
public class Contact implements Serializable{
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;
    @Column(name="firstName")
    private String firstName;
    @Column(name="lastName")
    private String lastName;
    @Column(name="email")
    private String email;
    @Column(name="phone")
    private String phone;
```

Bước 4: Tao ra một Session Bean ContactSessionBean , ContactSessionBeanRemote như sau :

```
@Remote
public interface ContactSessionBeanRemote {
    void addContact(Contact c);
    List listContact();
    Contact findOne(Long id);
    List findContact();
    List findByName(String name);
    Contact findByEmail(String email);
    Contact updateContact(Contact c);
    void deleteContact(Long id);
}
```

```
@Stateless
//@DeclareRoles({"admin", "user"})
public class ContactSessionBean implements ContactSessionBeanRemote {

    @PersistenceUnit(unitName = "ShoppingPU")
    private EntityManagerFactory entityManagerFactory;
    private EntityManager em;

    @Override
//    @RolesAllowed({"admin"})
    public void addContact(Contact c) {
        em = entityManagerFactory.createEntityManager();
//          em.loc
        em.persist(c);
    }

    @Override
//    @PermitAll
    public List listContact() {
        em = entityManagerFactory.createEntityManager();
        List<Contact> result = em.createNamedQuery("Contact.findAll", Contact.class).getResultList();
        return result;
    }
```

```
@Override
  @PermitAll
public List listContact() {
    em = entityManagerFactory.createEntityManager();
    List<Contact> result = em.createNamedQuery("Contact.findAll", Contact.class).getResultList();
    return result;
}

@Override
  @PermitAll
public Contact findOne(Long id) {
    em = entityManagerFactory.createEntityManager();
    return em.find(Contact.class, id);
}

@Override
  @PermitAll
public List findContact() {
    em = entityManagerFactory.createEntityManager();
    List<Contact> result = em.createNamedQuery("Contact.findAll", Contact.class).getResultList();
    return result;
}
```

```
@Override
  @PermitAll
public Contact findByEmail(String email) {
    em = entityManagerFactory.createEntityManager();
    Contact result = em.createNamedQuery("Contact.findByEmail", Contact.class).setParameter("email", email).getSingleResult();
    return result;
}

@Override
public Contact updateContact(Contact c) {
    em = entityManagerFactory.createEntityManager();
    Contact curr = em.find(Contact.class, c.getId());
    curr.setEmail(c.getEmail());
    curr.setFirstName(c.getFirstName());
    curr.setLastName(c.getLastName());
    curr.setPhone(c.getPhone());
    em.persist(curr);
    return curr;
}

@Override
  @DenyAll
public void deleteContact(Long id) {
    em = entityManagerFactory.createEntityManager();
    Contact curr = em.find(Contact.class, id);
    em.remove(curr);
}
```

Bước 5: Tạo ra 1 web client xây dựng giao diện them , xoá , xem danh sách danh dạ, tìm kiếm danh bạ:

```
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
      processRequest(request, response);
    List<Contact> listContact = contactSessionBeanRemote.listContact();
    request.setAttribute("listContact", listContact);

    RequestDispatcher rd = getServletContext().getRequestDispatcher("/home.jsp");
    rd.forward(request, response);
}
```

```
<div class="container-fluid">
    <div class="row">
        <a class="btn btn-primary" href="AddContact" role="button">Thêm mới</a>
    </div>
    <div class="row">
        <table class="table table-hover">
            <thead>
                <tr>
                    <th>ID</th>
                    <th>First Name</th>
                    <th> Last Name </th>
                    <th>Email</th>
                    <th>Control</th>
                </tr>
            </thead>
            <tbody>
            <c:forEach var="row" items="${requestScope.listContact}">
                <tr>
                    <td><c:out value="${row.id}"/></td>
                    <td><c:out value="${row.firstName}"/></td>
                    <td><c:out value="${row.lastName}"/></td>
                    <td><c:out value="${row.email}"/></td>
                    <td><c:out value="${row.phone}"/></td>
                    <td>
                        <a class="btn btn-warning" href="edit-trainee?id=<c:out value="${row.id}"/>" role="button">Edit</a>|||
                        <a class="btn btn-danger" href="delete-trainee?id=<c:out value="${row.id}"/>" role="button">Delete</a>
                    </td>
                </tr>
            </c:forEach>
        </tbody>
    </table>
</div>
```

```
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
      processRequest(request, response);
    RequestDispatcher rd = getServletContext().getRequestDispatcher("/create-contact.jsp");
    rd.forward(request, response);
}

/**
 * Handles the HTTP <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
      processRequest(request, response);
    Contact c = new Contact(request.getParameter("txtFirstName"),
            request.getParameter("txtLastName"), request.getParameter("txtEmail"), request.getParameter("txtphone"));
    contactSessionBeanRemote.addContact(c);
    response.sendRedirect(request.getContextPath() + "/ContactServlet");
}
```

```
<body>
    <div class="container-fluid">
        <div class="row">
            <form class="form-horizontal" action="" method="POST" >
                <div class="form-group">
                    <label for="txtTitle" class="col-sm-2 control-label">Firt Name:</label>
                    <div class="col-sm-10">
                        <input type="text" class="form-control" id="txtFirstName" name="txtFirstName" value="" required >
                    </div>
                </div>
                <div class="form-group">
                    <label for="txtAuthor" class="col-sm-2 control-label">Last Name:</label>
                    <div class="col-sm-10">
                        <input type="text" class="form-control" id="txtLastName" name="txtLastName" value="" required >
                    </div>
                </div>
                <div class="form-group">
                    <label for="txtDescription" class="col-sm-2 control-label">Email:</label>
                    <div class="col-sm-10">
                        <input type="email" class="form-control" id="txtEmail" name="txtEmail" value="" required >
                    </div>
                </div>
                <div class="form-group">
                    <label for="txtDescription" class="col-sm-2 control-label">Phone:</label>
                    <div class="col-sm-10">
                        <input type="text" class="form-control" id="txtEmail" name="txtphone" value="" required >
                    </div>
                </div>
                <div class="form-group">
                    <div class="col-sm-offset-2 col-sm-10">
                        <button type="submit" class="btn btn-default btnSubmitDiem">Luu lai</button>
                    </div>
                </div>
            </form>

        </div>
    </div>
</body>
```