**BÀI TẬP THỰC HÀNH TUẦN 1**

# Yêu cầu

Tạo một csdl có tên mydb với các bảng dữ liệu **account** (account\_id, full\_name, password, email, phone, status), bảng **role** (role\_id, role\_name, description, status). Một account thuộc về nhiều role, mỗi role có thể có nhiều account. Trong đó status là trường để chỉ trạng thái mẫu tin (1-active, 0-deactive, -1-xóa)

(Tip: tạo 1 bảng thứ ba có tên **grant\_access** (account\_id, role\_id, is\_grant, note) với trường is\_grant nhận giá trị 0-diasable, 1-enable.)

Một bảng có tên **log** để ghi vết lần đăng nhập. Thông tin bao gồm: id (auto-increment), account đăng nhập, ngày giờ đăng nhập, ngày giờ đăng xuất, ghi chú.

Tạo một jakartaEE project có tên **week01\_lab\_HotenSv\_mssv** (Có thể kết nối GitHub/GitLab để push code). Thực hiện các công việc sau:

* Tạo một servlet có tên **ControlServlet** (partern cùng tên). Servlet này nhận một tham số (parameter) có tên là **action**. Tham số này nhận các giá trị chuỗi để chỉ các hành động tương ứng.
* Kết nối với csdl, thực hiện các chức năng:
  + Thêm, cập nhật, xóa ở các bảng đã cho.
  + Đăng nhập
  + Hiển thị thông tin tài khoản (nếu đăng nhập thành công)
  + Hiển thị các quyền của một account.
  + Hiển thị các account của một role
  + Cấp quyền cho một account
  + Ghi log mỗi lần account đăng nhập, đăng xuất.
* Một trang html hiển thị cửa sổ đăng nhập. Nếu đăng nhập thành công và là quyền admin thì hiển thị trang dashboard cho phép quản lý các account khác (bao gồm các quyền thêm, xóa, sửa và cấp quyền). Còn không (không phải admin) thì hiển thị thông tin của người đăng nhập cùng các quyền mà người đó có.

Upload project lên Github/GitLab/BitBucket và chia sẻ code với guider để chấm điểm.

# Hướng dẫn thực hành:

Cấu trúc của project có dạng Figure 1.

Trong Controller Servlet, ở các phương thức (GET, POST,…) ta lấy tham số action bằng cách

|  |
| --- |
| String action = request.getParameter("action"); |

Sau đó tùy thuộc action mà xử lý công việc khác nhau.

Client driver cho MariaDB:

|  |
| --- |
| Gradle:  implementation 'org.mariadb.jdbc:mariadb-java-client:3.2.0' |
| Maven:  <dependency>  <groupId>**org.mariadb.jdbc**</groupId>  <artifactId>**mariadb-java-client**</artifactId>  <version>**3.2.0**</version>  </dependency> |

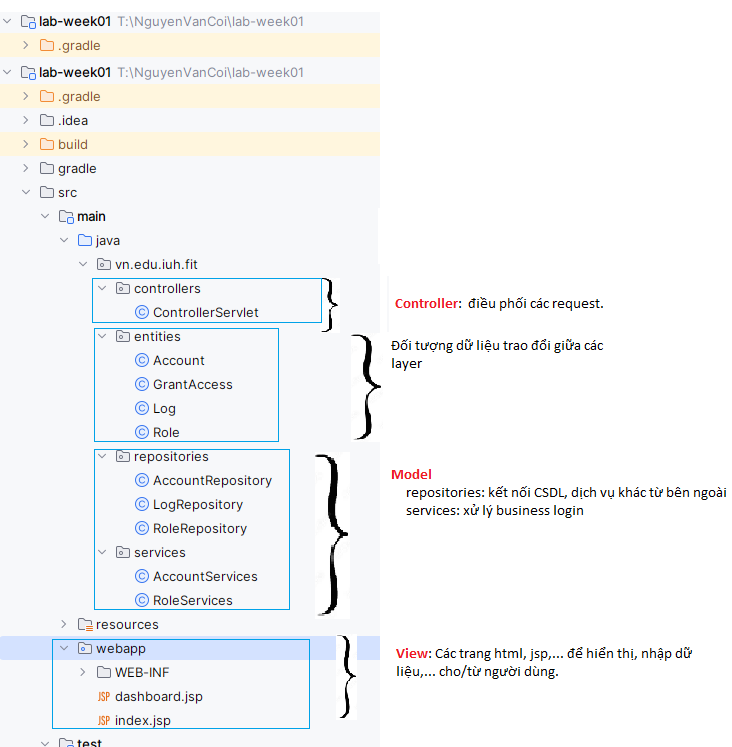
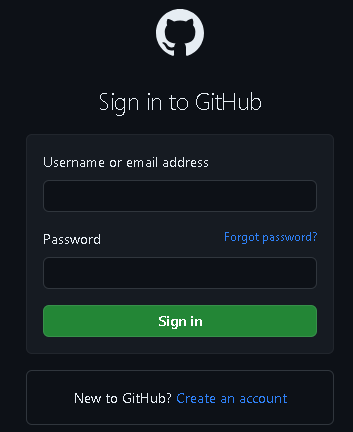


Figure 1. Project structure

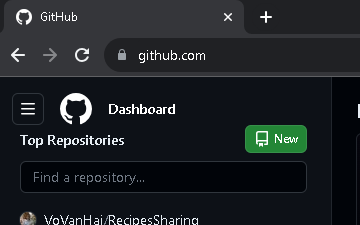
# Hướng dẫn Git-Hub:

Vào trang : <https://github.com/login>

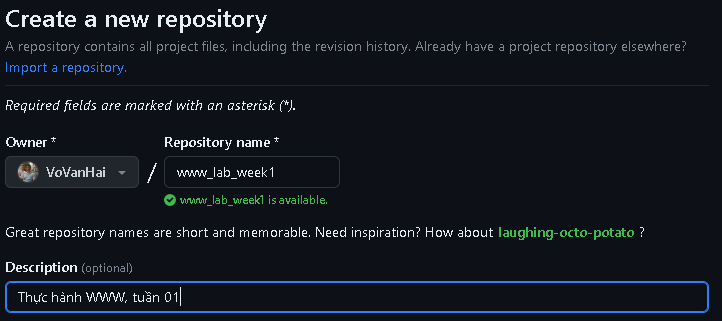


Đăng nhập vào Git-Hub. Nếu chưa có tài khoản thì tạo tài khoản mới.

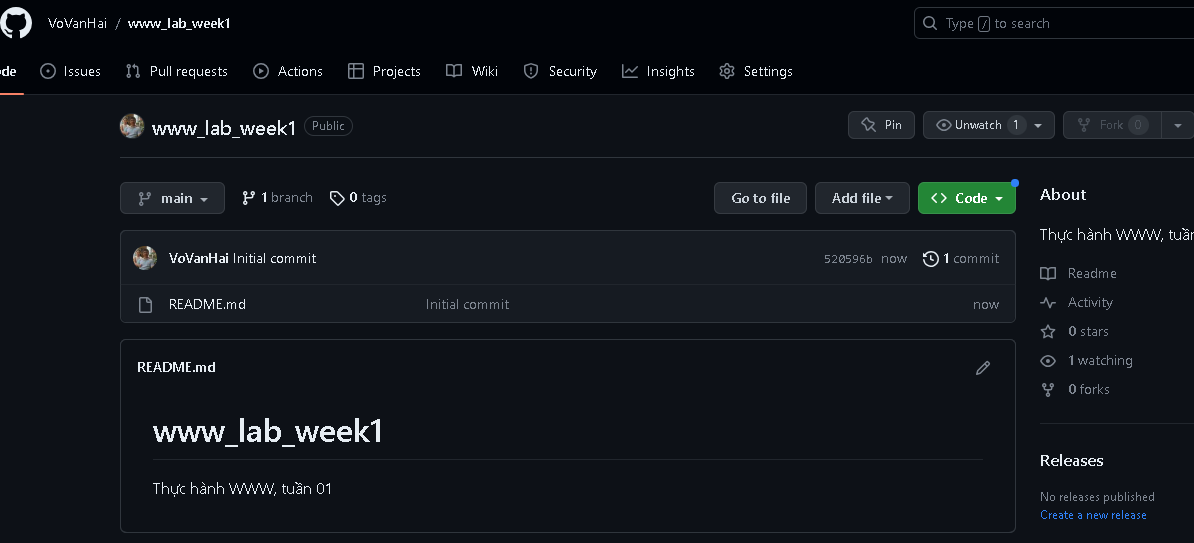
Sau khi đăng nhập thành công, cửa sổ sẽ ở dạng này



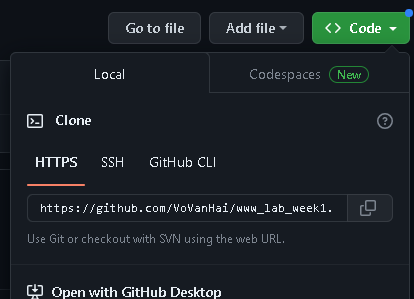
Nhấn New để tạo mới 1 project



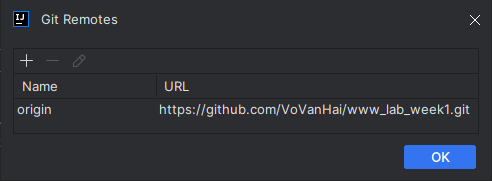
Nhấn nút Create bên dưới để tạo project

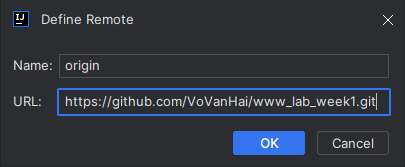


Nhấn nút Code, copy URL.

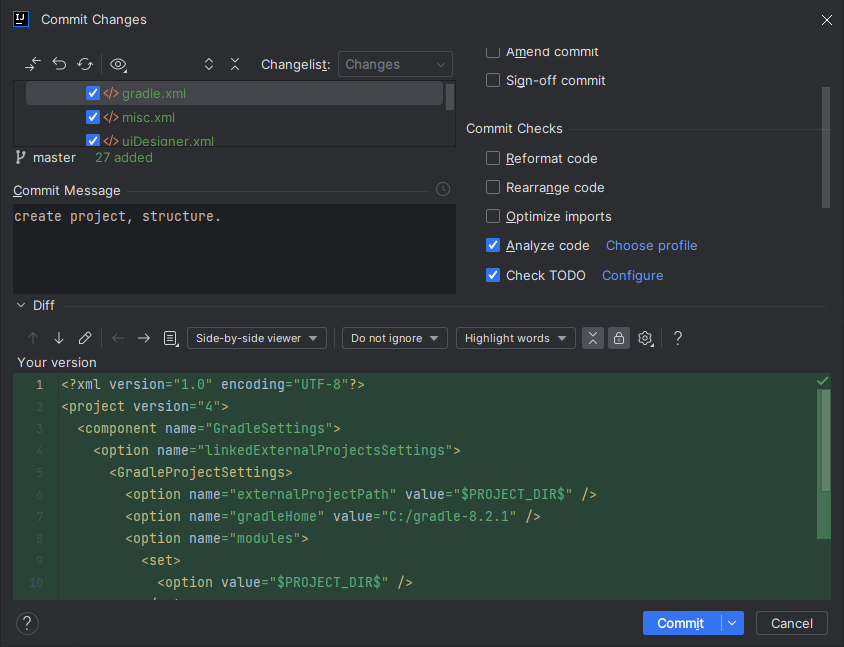


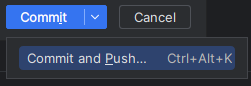
Trên IntelliJ IDEA, vào menu Git (sẽ xuất hiện nếu ban đầu bạn check vào  khi tạo mới project), chọn item “Manage Remotes…”. Một cửa sổ xuất hiện. Nhấn nút + để thêm vào một remote git repository, như hình



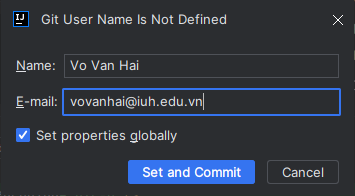


Vào lại menu Git, chọn Commit, cửa sổ Commit Changes xuất hiện. Nhập vào Commit Message

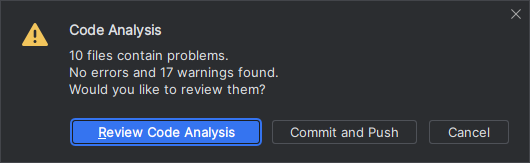


rồi nhấn xổ xuống nhỏ bên cạnh nút Commit . Chọn Commit and Push. (hoặc nhấn Commit rồi sau đó vào menu Git nhấn Push).

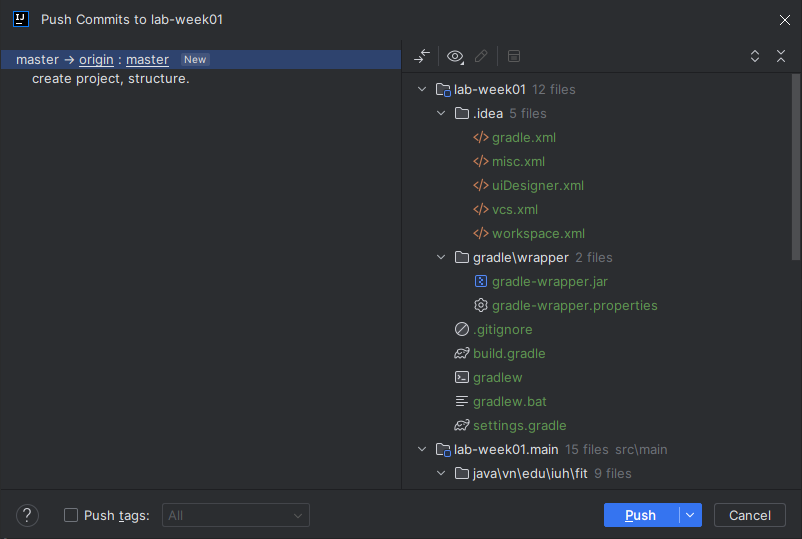
Nếu bạn dev trên máy của trường, sẽ có cửa sổ xuất hiện cho việc nhập tên, email của người commit. Nhập thông tin và nhấn Set and Commit



Git sẽ đánh giá code và đưa cảnh báo nếu code của bạn còn warning hay error. Nếu cần check lại thì nhấn nút “Review Code Analysis”, nếu cứ chọn commit thì nhấn nút “Commit and Push”. Trong trường hợp thử nghiệm này, “Commit and Push” sẽ được chọn. Khuyến cáo nên chọn Review Code Analysis.

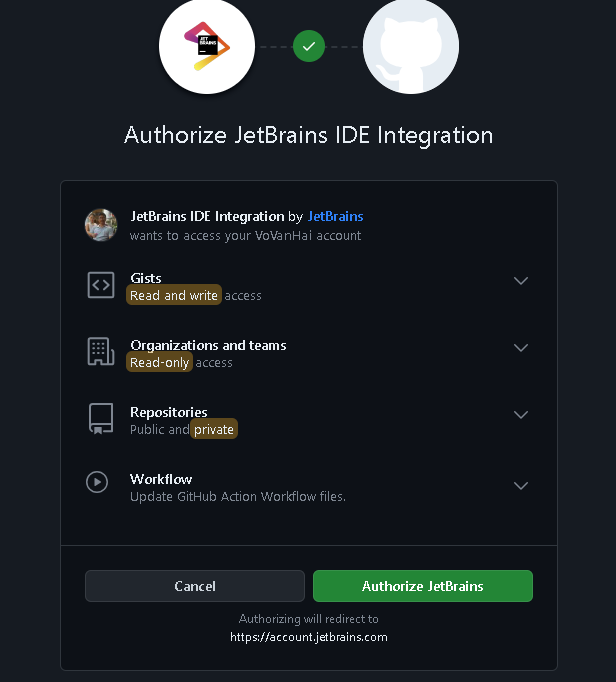


Sau khi Commit, cửa sổ Push sẽ xuất hiện

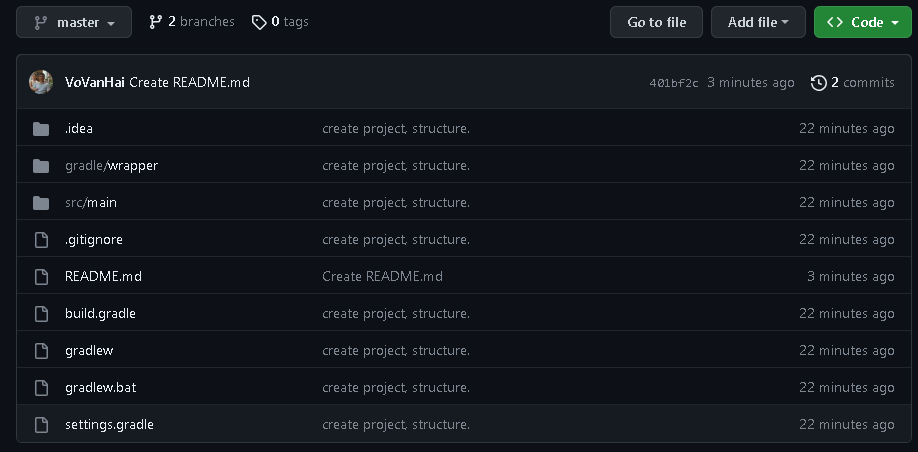


Mọi thứ OK thì bạn nhấn Push để đẩy lên GitHub.

Bạn có lẽ sẽ được nhận 1 khuyến cáo Authorize để IntelliJcó thể tích hợp với Git. Bạn nên đồng ý như sau



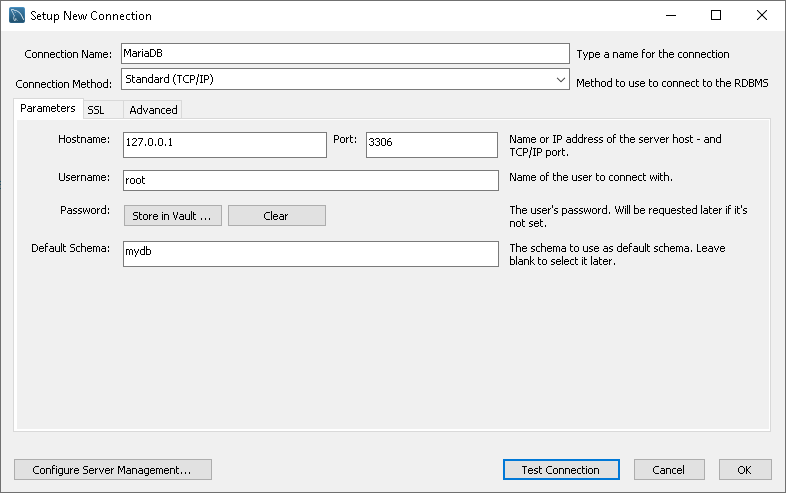
Sau khi commit, bạn sẽ thấy thay đổi trên GitHub



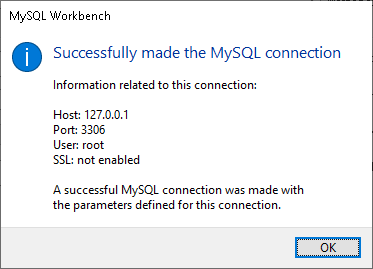
# Database script

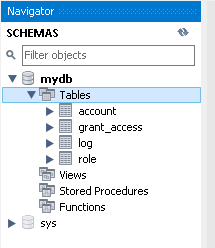
|  |
| --- |
|  |
| -- Dumping database structure for mydb  **CREATE** **DATABASE** **IF** **NOT** **EXISTS** `mydb` /\*!40100 DEFAULT CHARACTER SET latin1 COLLATE latin1\_swedish\_ci \*/**;**  **USE** `mydb`**;**  -- Dumping structure for table mydb.account  **CREATE** **TABLE** **IF** **NOT** **EXISTS** `account` **(**  `account\_id` varchar**(**50**)** **NOT** **NULL,**  `full\_name` varchar**(**50**)** **NOT** **NULL,**  `password` varchar**(**50**)** **NOT** **NULL,**  `email` varchar**(**50**)** **DEFAULT** **NULL,**  `phone` varchar**(**50**)** **DEFAULT** **NULL,**  `status` tinyint**(**4**)** **NOT** **NULL** **DEFAULT** 1**,**  **PRIMARY** **KEY** **(**`account\_id`**)**  **)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**latin1 **COLLATE=**latin1\_swedish\_ci**;**  -- Dumping data for table mydb.account: ~2 rows (approximately)  **INSERT** **INTO** `account` **(**`account\_id`**,** `full\_name`**,** `password`**,** `email`**,** `phone`**,** `status`**)** **VALUES**  **(**'met'**,** 'Tran Thi Met'**,** '123'**,** 'met@gmail.com'**,** '0904567890'**,** 1**),**  **(**'teo'**,** 'NGuyen Van Teo'**,** '123'**,** 'teo@gmail.com'**,** '0903123456'**,** 1**);**  -- Dumping structure for table mydb.grant\_access  **CREATE** **TABLE** **IF** **NOT** **EXISTS** `grant\_access` **(**  `role\_id` varchar**(**50**)** **NOT** **NULL,**  `account\_id` varchar**(**50**)** **NOT** **NULL,**  `is\_grant` bit**(**1**)** **NOT** **NULL** **DEFAULT** b'1'**,**  `note` varchar**(**250**)** **DEFAULT** ''**,**  **PRIMARY** **KEY** **(**`role\_id`**,**`account\_id`**),**  **KEY** `account\_grant` **(**`account\_id`**),**  **CONSTRAINT** `account\_grant` **FOREIGN** **KEY** **(**`account\_id`**)** **REFERENCES** `account` **(**`account\_id`**)** **ON** **DELETE** **CASCADE** **ON** **UPDATE** **CASCADE,**  **CONSTRAINT** `role\_grant` **FOREIGN** **KEY** **(**`role\_id`**)** **REFERENCES** `role` **(**`role\_id`**)** **ON** **DELETE** **CASCADE** **ON** **UPDATE** **CASCADE**  **)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**latin1 **COLLATE=**latin1\_swedish\_ci**;**  -- Dumping data for table mydb.grant\_access: ~2 rows (approximately)  **INSERT** **INTO** `grant\_access` **(**`role\_id`**,** `account\_id`**,** `is\_grant`**,** `note`**)** **VALUES**  **(**'admin'**,** 'teo'**,** b'1'**,** ''**),**  **(**'user'**,** 'met'**,** b'1'**,** ''**);**  -- Dumping structure for table mydb.log  **CREATE** **TABLE** **IF** **NOT** **EXISTS** `**log**` **(**  `id` bigint**(**20**)** **NOT** **NULL** AUTO\_INCREMENT**,**  `account\_id` varchar**(**50**)** **NOT** **NULL,**  `login\_time` datetime **NOT** **NULL** **DEFAULT** **current\_timestamp(),**  `logout\_time` datetime **NOT** **NULL** **DEFAULT** **current\_timestamp(),**  `notes` varchar**(**250**)** **NOT** **NULL** **DEFAULT** ''**,**  **PRIMARY** **KEY** **(**`id`**)**  **)** ENGINE**=**InnoDB AUTO\_INCREMENT**=**2 **DEFAULT** CHARSET**=**latin1 **COLLATE=**latin1\_swedish\_ci **COMMENT=**'ghi logs'**;**  -- Dumping data for table mydb.log: ~0 rows (approximately)  -- Dumping structure for table mydb.role  **CREATE** **TABLE** **IF** **NOT** **EXISTS** `role` **(**  `role\_id` varchar**(**50**)** **NOT** **NULL,**  `role\_name` varchar**(**50**)** **NOT** **NULL,**  `description` varchar**(**50**)** **DEFAULT** **NULL,**  `status` tinyint**(**4**)** **NOT** **NULL,**  **PRIMARY** **KEY** **(**`role\_id`**)**  **)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**latin1 **COLLATE=**latin1\_swedish\_ci**;**  -- Dumping data for table mydb.role: ~2 rows (approximately)  **INSERT** **INTO** `role` **(**`role\_id`**,** `role\_name`**,** `description`**,** `status`**)** **VALUES**  **(**'admin'**,** 'administrator'**,** 'admin role'**,** 1**),**  **(**'user'**,** 'user'**,** 'user role'**,** 1**);** |

Muốn xem relationship diagram: vào trang <https://dev.mysql.com/downloads/workbench/>, download MySQL Workbench (MW) sau đó cài đặt sau đó mở MySQL Workbench lên.

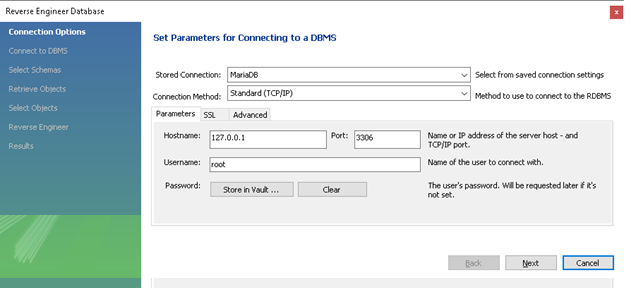


Tạo 1 kết nối. Nhấn Test Connection để kiểm tra kết nối thành công không.





Trong MW, Vào menu Database, chọn “**Reverse Engineer**”.



Nhấn Next cho đến khi hoàn tất.

ER diagram sẽ xuất hiện và bạn có thể hiệu chỉnh nó.

