# Advanced Web Programming
# Eloquent ORM

**Name**
Dicha Zelianivan Arkana

**NIM**
2241720002

**Class**
2i

**Department**
Information Technology

**Study Program**
D4 Informatics Engineering

## 1 Practicum 1

3. It creates a new record in the database with the given data. We use the *mass assignment* feature to insert the data into the database.

6. It throws an error because we do not include the `password` field into the `$fillable` property in the `UserModel` model.

## 2 Practicum 2.1

3. It now returns the user with the `id` of 1 and display it in the browser.

5. It finds the user with the `level_id` of 1 and fetch the first result from the database and return it to the view.

7. It uses an alternative way to find the user with the `level_id` of 1 by using the `firstWhere` which is basically a combination of `where` and `first` method.

9. It returns 404 page because it didn't find any user with the `username` or `nama` of `1` so it returns a fallback view which we defined as 404 page by doing `abort(404)`.

11. It returns the same result as the previous one because it didn't find any user with the `username` or `nama` of `20`

## 3 Practicum 2.2

2. It returns the first user with the ID of 1 and display it in the browser.

4. It returns a 404 page because we didn't find any user with the `username` of `manager9`

## 4 Lab Course 2.3

2. It returns the debug output of the count result of the query.

## 5 Practicum 2.4

5. It creates the user if it doesn't exist in the database, otherwise it will update the user with the given data.

7. It actually depends on the data that we have, it will return the user with the `username` of `manager` if it exists, otherwise it will create a new user with the given data. Although, it will throw an error since we don't include the rest of the data

9. It will return the user if it already exists, otherwise it will create a new user with the data that we've provided but not persist it in the database.

11. It will return the user with the `username` of `manager` if it exists, otherwise it will create a new user with the given data and now it will persist it in the database.

## 6 Practicum 2.5

4. It will return `true` to the browser since we modified the field.

## 7 Practicum 2.6

4. It creates a route that will render the page filled with a table populated by the users data from the database.

8. It will navigate to the add user page because we added the route for that and we set on the user list page that the anchor tag will allow us to navigate to the add user page.

11. It will now create a user based on the data that we've given. After the data is created, we are redirected to the `/user` route.

14. It will create a page with a form in which we can update the user data later on, but for now we haven't added the route handler for updating the user data.

17. It will update the user data from the data that we've given using the find method and then changing its attribute, and then finally persist the update to the database using the `save` method.

19. It will delete the user when clicked because we added the remove user route handler, which wil use the delete method to delete the user from the database.

## 8 Practicum 2.7

3. It outputs the user along with its level since we defined the relation and include it in the query

6. It outputs a table filled with the user data along with its level data since we added them to the table.

The code can be found in `https://github.com/elianiva/pwl_pos`