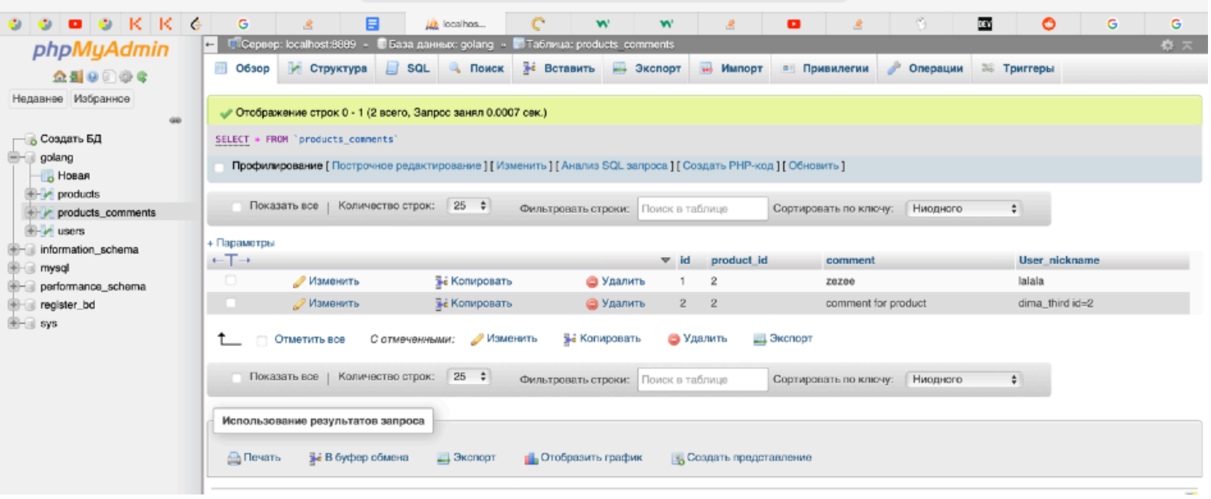
On this week, we implemented comments for each product,
first of all we created new table for comments,



✓ Отображение строк 0 - 1 (2 всего, Запрос занял 0.0007 сек.)

```
SELECT * FROM `products_comments`
```

□ Профилирование [Построчное редактирование] [Изменить] [Анализ SQL запроса] [Создать PHP-код] [Обновить]

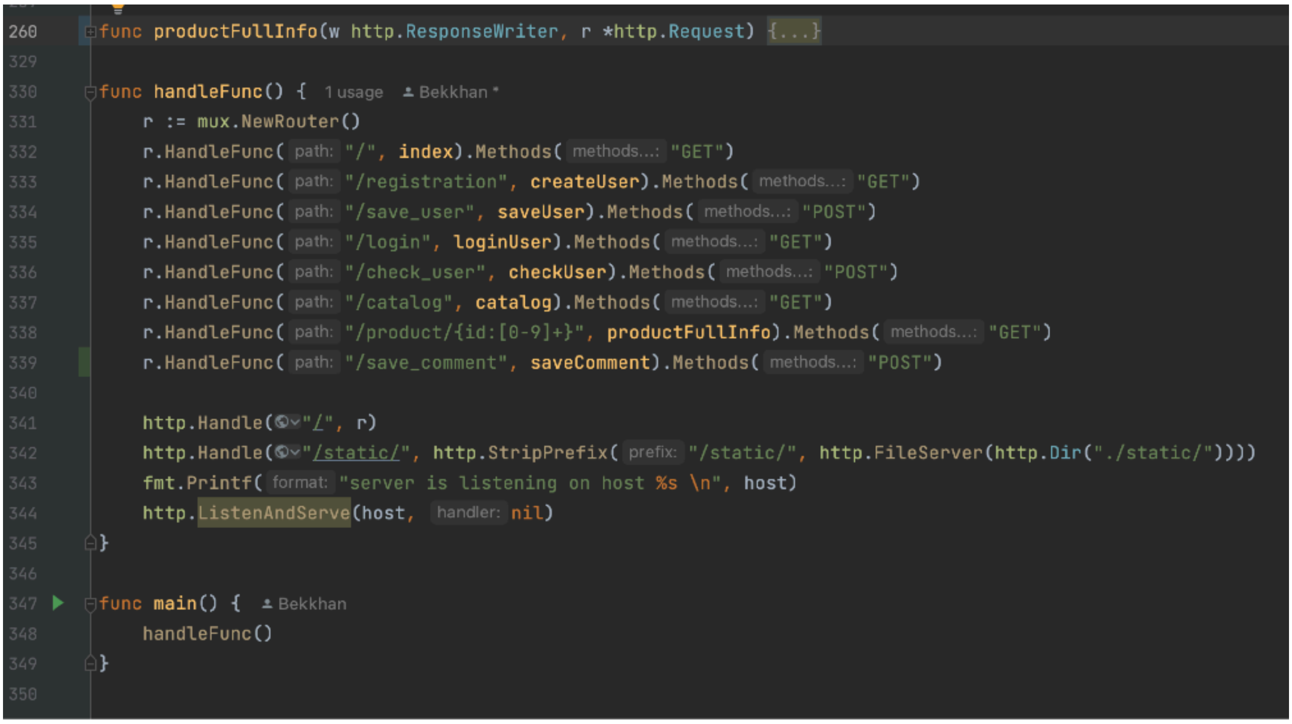
□ Показать все | Количество строк: 25 Фильтровать строки: Поиск в таблице Сортировать по ключу: Ниодного

+ Параметры

				id	product_id	comment	User_nickname
□	Изменить	Копировать	Удалить	1	2	zezee	lalala
□	Изменить	Копировать	Удалить	2	2	comment for product	dima_third id=2

□ Отметить все
С отмеченными:
Изменить
Копировать
Удалить
Экспорт
□ Показать все | Количество строк: 25
Фильтровать строки: Поиск в таблице
Сортировать по ключу: Ниодного
Использование результатов запроса
Печать
В буфер обмена
Экспорт
Отобразить график
Создать представление

Here u can see part of all code:
We will change productFullInfo function, which connecting with path of each product:



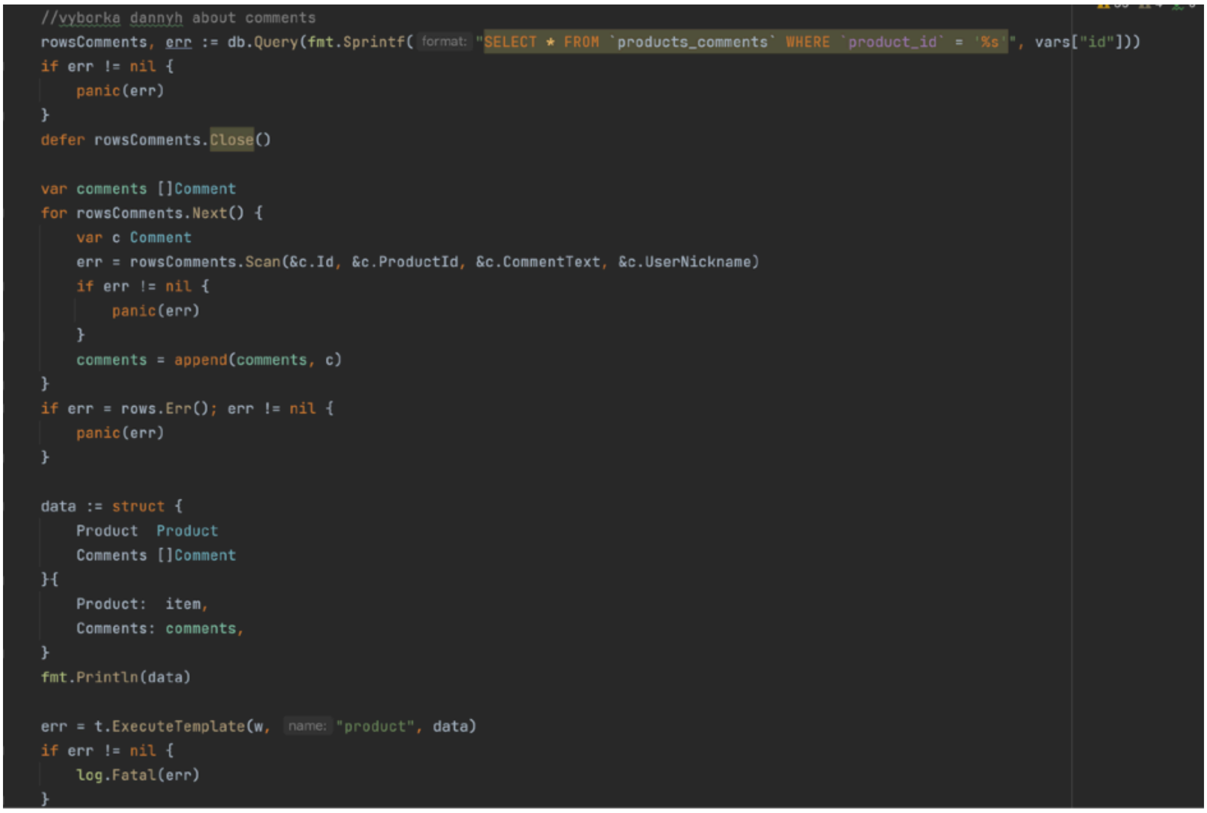
```
func productFullInfo(w http.ResponseWriter, r *http.Request) {...}

func handleFunc() {  1 usage  Bekkhan *
    r := mux.NewRouter()
    r.HandleFunc( path: "/", index).Methods( methods...: "GET")
    r.HandleFunc( path: "/registration", createUser).Methods( methods...: "GET")
    r.HandleFunc( path: "/save_user", saveUser).Methods( methods...: "POST")
    r.HandleFunc( path: "/login", loginUser).Methods( methods...: "GET")
    r.HandleFunc( path: "/check_user", checkUser).Methods( methods...: "POST")
    r.HandleFunc( path: "/catalog", catalog).Methods( methods...: "GET")
    r.HandleFunc( path: "/product/{id:[0-9]+}", productFullInfo).Methods( methods...: "GET")
    r.HandleFunc( path: "/save_comment", saveComment).Methods( methods...: "POST")

    http.Handle("/", r)
    http.Handle("/static/", http.StripPrefix( prefix: "/static/", http.FileServer(http.Dir("./static/"))))
    fmt.Printf( format: "server is listening on host %s \n", host)
    http.ListenAndServe(host,  handler: nil)
}

func main() {  Bekkhan
    handleFunc()
}
```

Here, u can see part, where data from mysql table is placing into the struct data, and then we executing template with that struct
We created struct data, because we should execute template product with multiple data, first - product, second - all comments of this product:



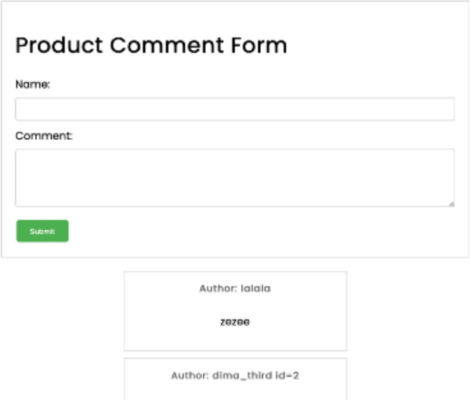
```
//vyborka dannyh about comments
rowsComments, err := db.Query(fmt.Sprintf( format: "SELECT * FROM `products_comments` WHERE `product_id` = '%s'", vars["id"]))
if err != nil {
    panic(err)
}
defer rowsComments.Close()

var comments []Comment
for rowsComments.Next() {
    var c Comment
    err = rowsComments.Scan(&c.Id, &c.ProductId, &c.CommentText, &c.UserNickname)
    if err != nil {
        panic(err)
    }
    comments = append(comments, c)
}
if err = rows.Err(); err != nil {
    panic(err)
}

data := struct {
    Product  Product
    Comments []Comment
}{
    Product:  item,
    Comments: comments,
}
fmt.Println(data)

err = t.ExecuteTemplate(w,  name: "product", data)
if err != nil {
    log.Fatal(err)
}
```

Below u can see form for comment, its comments and html code of form :



Product Comment Form

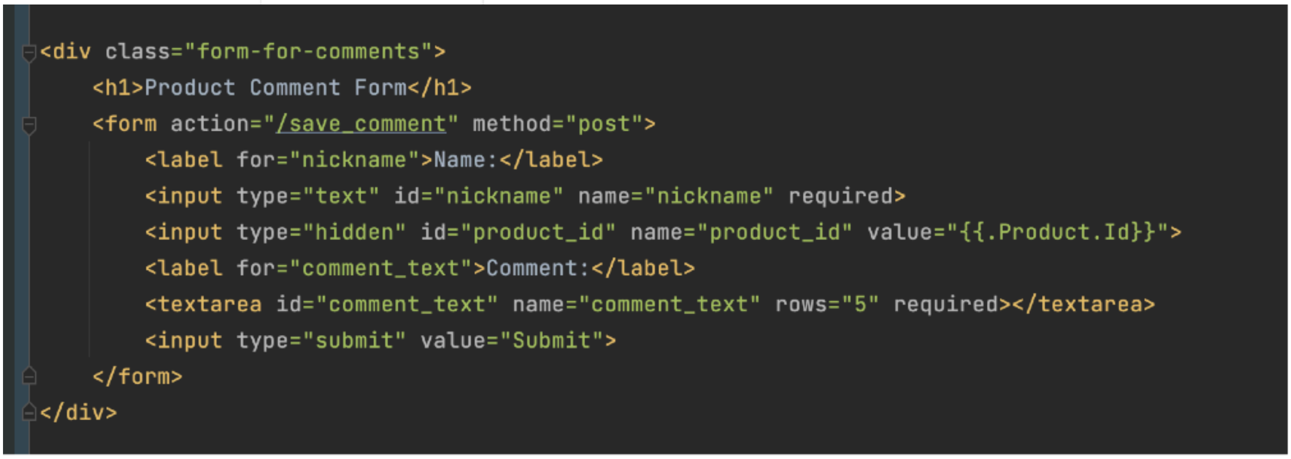
Name:

Comment:

Author: lalala

zezee

Author: dima_third id=2



```
<div class="form-for-comments">
    <h1>Product Comment Form</h1>
    <form action="/save_comment" method="post">
        <label for="nickname">Name:</label>
        <input type="text" id="nickname" name="nickname" required>
        <input type="hidden" id="product_id" name="product_id" value="{{.Product.Id}}">
        <label for="comment_text">Comment:</label>
        <textarea id="comment_text" name="comment_text" rows="5" required></textarea>
        <input type="submit" value="Submit">
    </form>
</div>
```