CS 510: Rust Programming
Registration Form Project Whitepaper
Haomin He

Overview

This project is aimed at creating an event registration form that helps event host knows more about his/her guests. For instance, how many people are coming to the event and how much vegetarian food should be ordered. I build a Wedding Registration Form that includes a front-end web interface in HTML, JavaScript, CSS, jQuery and Bootstrap, and a back-end server, I use Rust Rocket, Serde, and Diesel for server computation and communication.

Structure

In this project, there are four websites display useful information. They are: index.html, confirmation.html, guests.html, and localhost:8000/registered. The first two websites are used by guests for registering the event. The last two websites are used by host to display information of guests.

In index.html, there is a registering form for guests. We use guest's email as primary key to uniquely identify which guest is registered. We assume email addresses are distinct. There are no two guests share the same email address. Therefore we avoid the case that multiple guests have exact same full name. Event attender could be a vegetarian or a kid, so there are 'Vegetarian' and 'Sit At Kids Table' options (they need to answer Yes or No in the option textboxes). Guests can click on 'Submit' button after filling out the form. The page jumps to confirmation.html which shows they have registered the event successfully.

On the other side, host can see guests' information on guests.html, and localhost:8000/registered. The primary key email address is displayed first. Then it is followed by guest name, dietary and sitting table preferences. This display method helps host easily identify who the guests are under each different email address. Host can contact guests through emailing. Total number of guests is calculated at the end of guest.html page. If there are newly added guests, refreshing page guests.html can reflect new changes.

Object Relational Mapping framework, Diesel, is utilized for building guests database and making data persistent. The very first thing my program does is reading data from guests database; parse and display data on guests.html, and localhost:8000/registered. If there is nothing in the database yet, the program will display empty table. Newly added guest information is written to the database. As a result, if I close my current running program session, all my data is reserved. Next time I run my program again, my old guest data is displayed as well.

Conclusion

I have learned a lot from this project. At the beginning, I thought Rust back-end would be much harder compare to Node.js. But actually, as soon as I learned how to use Rocket, Serde, and Diesel frameworks; creating website server in Rust became less challenging and understandable. Certainly, this requires lots of research and self-study. One mistake that I have made often is trying to use logics from another language on Rust. But practically, Rust frameworks can help me solve the problem much quicker and simpler.

References of this Project

https://stackoverflow.com/
https://github.com/diesel-rs/diesel/tree/master/diesel_cli
https://github.com/rust-lang-nursery/rustup.rs
https://github.com/diesel-rs/diesel/issues/1700
https://github.com/rust-lang/rust/issues/50825
https://medium.com/sean3z/building-a-restful-crud-api-with-rust-1867308352d8
https://github.com/sean3z/rocket-diesel-rest-api-example
https://github.com/diesel-rs/diesel/issues/321
http://www.goldsborough.me/rust/web/tutorial/2018/01/20/17-01-11-writing_a_microservice_in_rust/
https://github.com/SergioBenitez/Rocket/tree/v0.3.12/examples/json
https://api.rocket.rs/rocket_contrib/enum.Value.html
https://github.com/pwfantasy/pwfantasy-api
https://github.com/blackbeam/rust-mysql-simple
https://github.com/diesel-rs/diesel/issues/1730
http://spejss.com/index.php/2017/12/06/how-to-use-diesel-orm-in-rust/
https://rocket.rs/guide/state/