编号：

2020 - 2021学年第 二 学期



**实 验 报 告**

实验课程名称 软件架构与应用开发

专 业 班 级 计算机1803

学 生 姓 号 31801150

学 生 姓 名 张帅

实验指导教师 彭彬

一. 实验目的和要求

**实验要求：**

1.从零模仿实现L04对应的三个示例

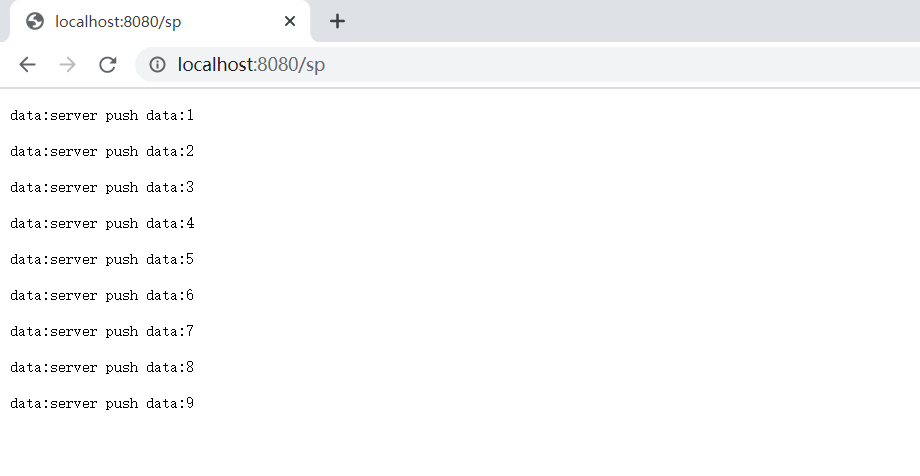
2. 结合课本WebFluxMongodb示例，修改实验三中学生基础信息接口服务，实现学生信息在MongoDB中的增删改查询操作

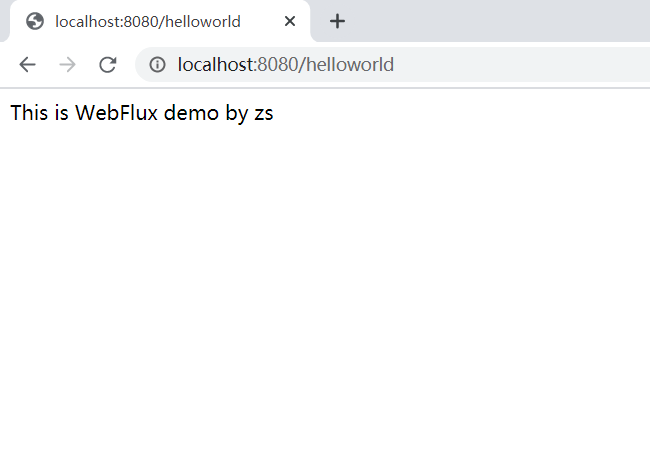
二. 实验内容

1. 从零模仿实现L04对应的三个示例

示例1：

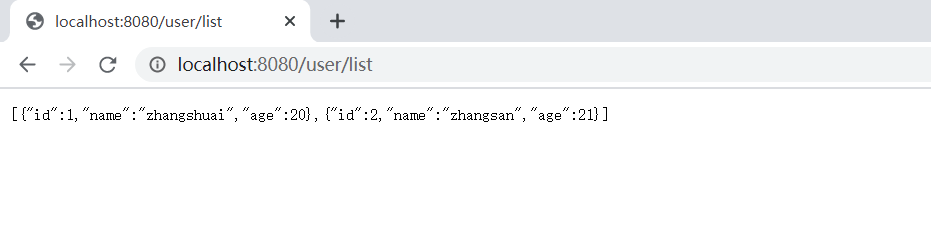
@RestController  
public class FluxController {  
 @GetMapping("/helloworld")  
 public Mono<String> helloworld() {  
 return Mono.*just*("This is WebFlux demo by zs");  
 }:  
 @GetMapping(value = "/sp", produces = MediaType.*TEXT\_EVENT\_STREAM\_VALUE*)  
 private Flux<String> serverPush() {  
 Flux<String> result = Flux  
 .*fromStream*(IntStream.*range*(1, 10).mapToObj(i -> {  
 try {  
 TimeUnit.*SECONDS*.sleep(1);  
 } catch (InterruptedException e) {  
 }  
 return "server push data:" + i;  
 }));  
 return result;  
 }  
}

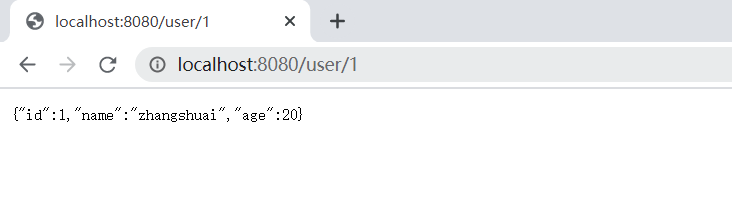


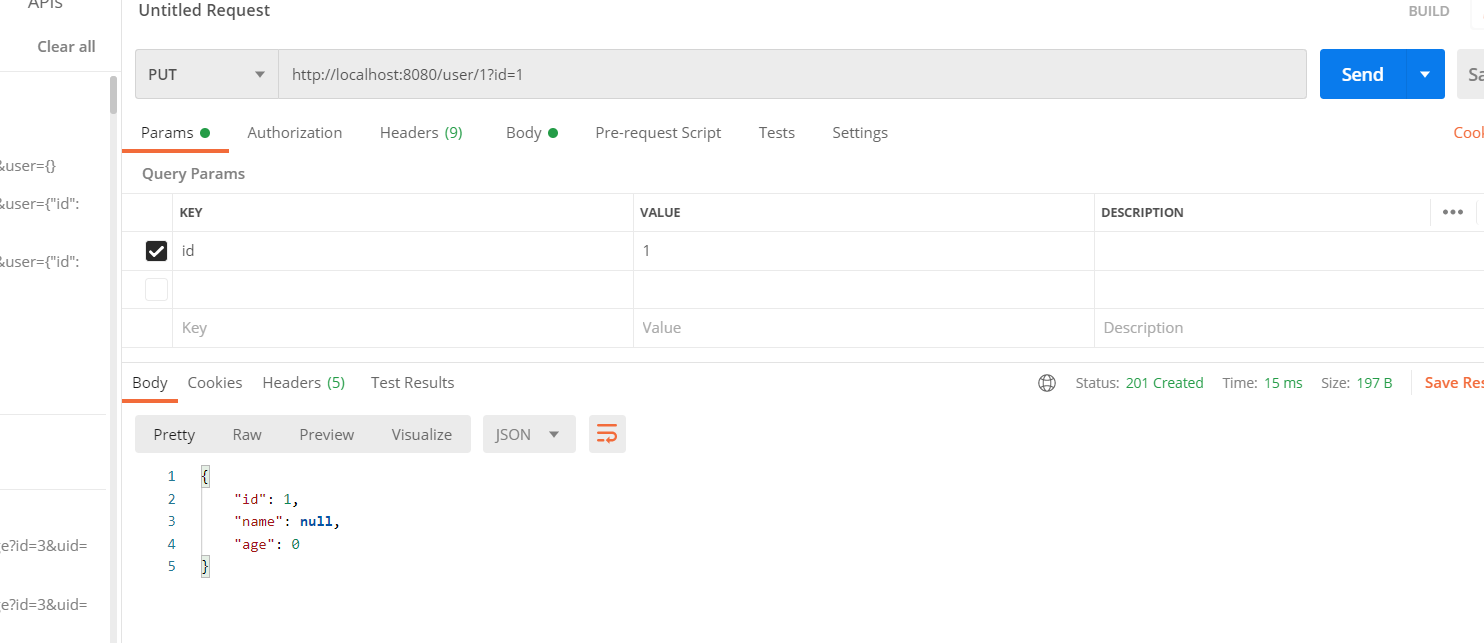


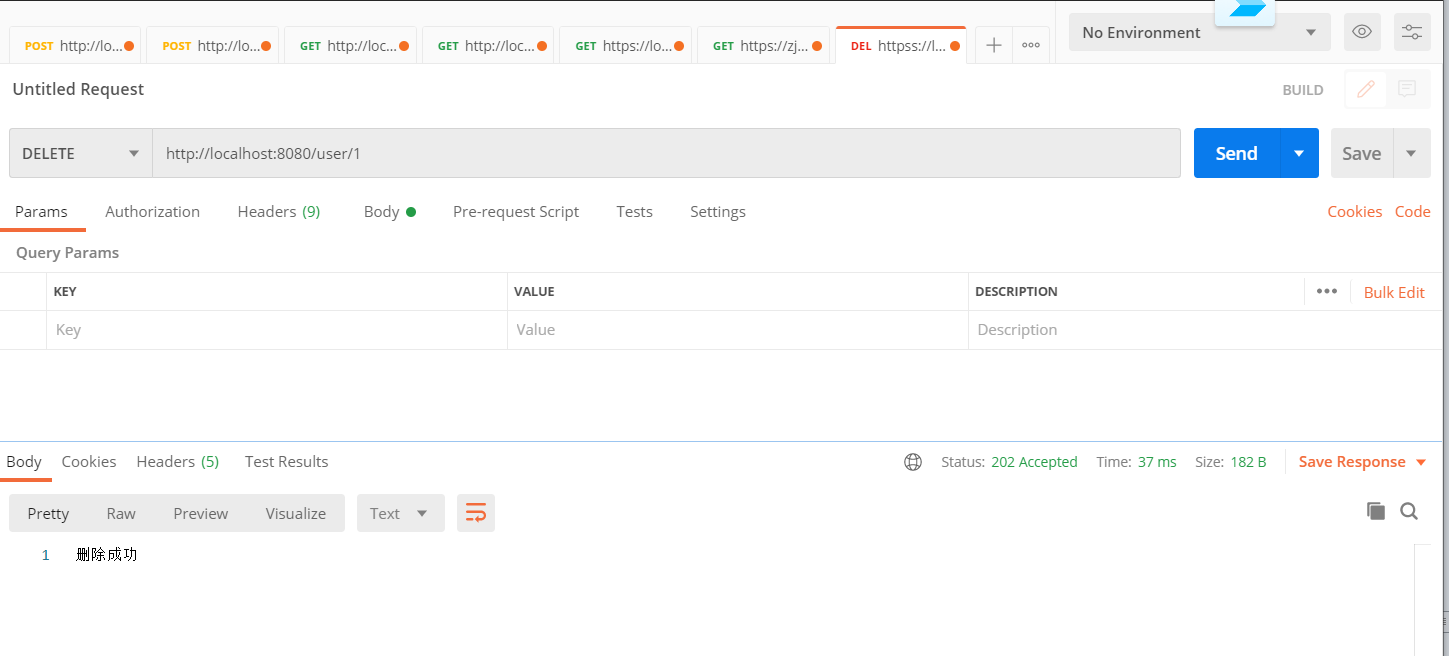
示例2：

@RestController  
@RequestMapping(path = "/user")  
public class UserController {  
 Map<Long, MVCUser> users = new HashMap<>();  
  
 @PostConstruct  
 public void init() throws Exception {  
 users.put(Long.*valueOf*(1), new MVCUser(1, "zhangshuai", 20));  
 users.put(Long.*valueOf*(2), new MVCUser(2, "zhangsan", 21));  
 }  
  
  
 @GetMapping("/list")  
 public Flux<MVCUser> getAll() {  
 return Flux.*fromIterable*(users.entrySet().stream()  
 .map(entry -> entry.getValue())  
 .collect(Collectors.*toList*()));  
 }  
  
  
 @GetMapping("/{id}")  
 public Mono<MVCUser> getUser(@PathVariable Long id) {  
 return Mono.*justOrEmpty*(users.get(id));  
 }  
  
  
 @PostMapping("")  
 public Mono<ResponseEntity<String>> addUser(MVCUser user) {  
 users.put(user.getId(), user);  
 return Mono.*just*(new ResponseEntity<>("添加成功", HttpStatus.*CREATED*));  
 }  
  
  
 @PutMapping("/{id}")  
 public Mono<ResponseEntity<MVCUser>> putUser(@PathVariable Long id, MVCUser user) {  
 user.setId(id);  
 users.put(id, user);  
 return Mono.*just*(new ResponseEntity<>(user, HttpStatus.*CREATED*));  
 }  
  
 @DeleteMapping("/{id}")  
 public Mono<ResponseEntity<String>> deleteUser(@PathVariable Long id) {  
 users.remove(id);  
 return Mono.*just*(new ResponseEntity<>("删除成功", HttpStatus.*ACCEPTED*));  
 }  
}





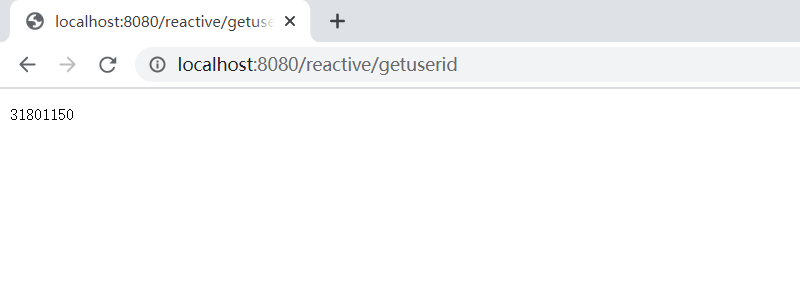
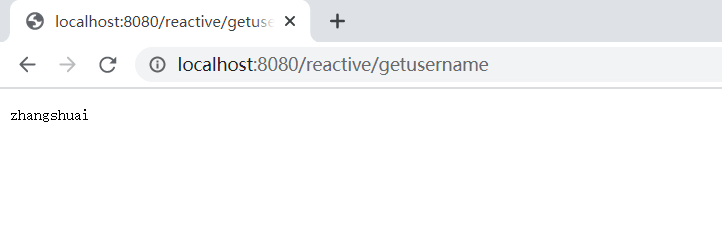


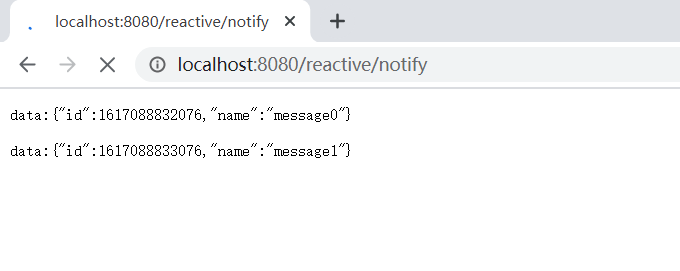


示例3：

@Component  
public class ReactiveUserHandler {  
  
 public Mono<ServerResponse> getUserId(ServerRequest serverRequest) {  
 return ServerResponse.*ok*()  
 .contentType(*APPLICATION\_JSON*)  
 .body(Mono.*just*("31801150"), String.class);  
 }  
  
  
 public Mono<ServerResponse> getUserName(ServerRequest serverRequest) {  
 return ServerResponse.*ok*().contentType(*APPLICATION\_JSON*).body(Mono.*just*("zhangshuai"), String.class);  
 }  
  
  
 public Mono<ServerResponse> notifyNewUser(ServerRequest serverRequest) {  
 return ServerResponse.*status*(HttpStatus.*OK*)  
 .contentType(MediaType.*TEXT\_EVENT\_STREAM*)  
 .body(Flux.*interval*(Duration.*ofSeconds*(1)).  
 map(s -> new ReactiveUser(System.*currentTimeMillis*() + s, "message" + s)), ReactiveUser.class);  
 }  
}

@Configuration  
public class ReactiveRouter {  
 @Autowired  
 private ReactiveUserHandler userHandler;  
 @Bean  
 public RouterFunction<ServerResponse> userRouter() {  
 return RouterFunctions.*nest*(  
  
 *path*("/reactive"),  
  
 RouterFunctions  
 .*route*(*GET*("/getuserid"), userHandler::getUserId)  
 .andRoute(*GET*("/getusername"), userHandler::getUserName)  
 .andRoute(*GET*("/notify"), userHandler::notifyNewUser)  
 );  
 }  
}

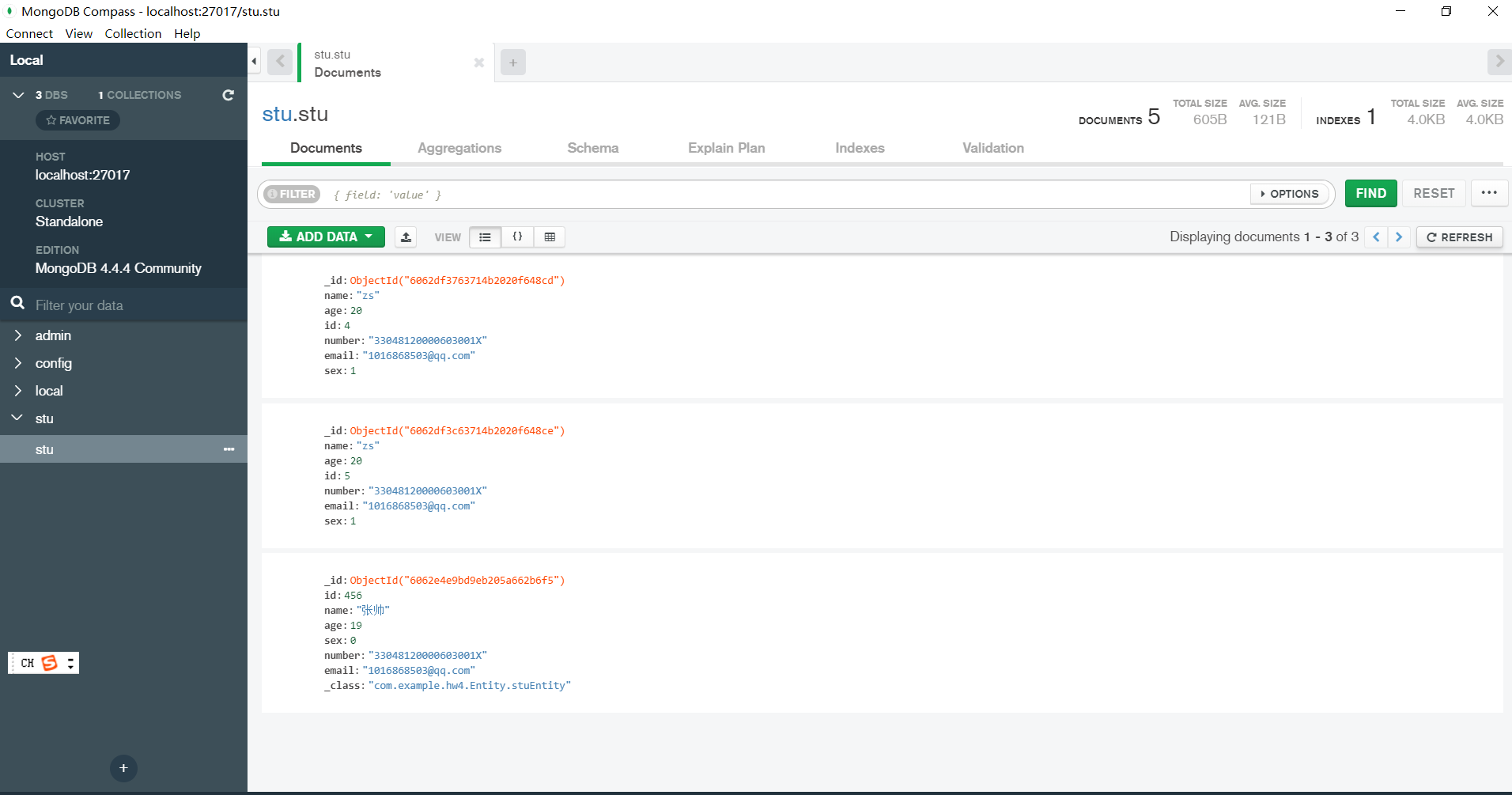
 



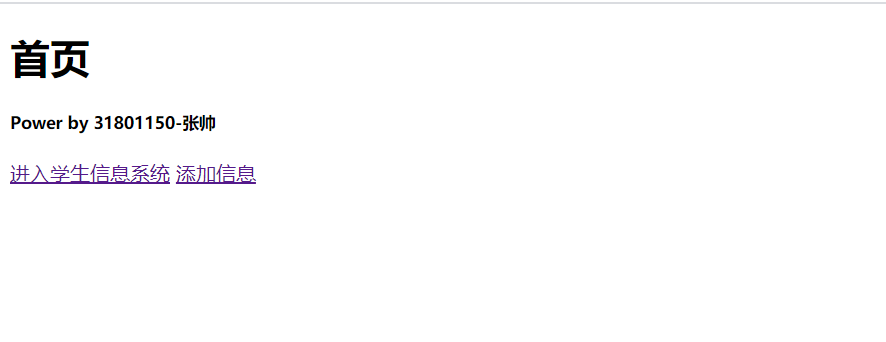
二、结合课本WebFluxMongodb示例，修改实验三中学生基础信息接口服务，实现学生信息在MongoDB中的增删改查询操作

代码见文件夹：hw4

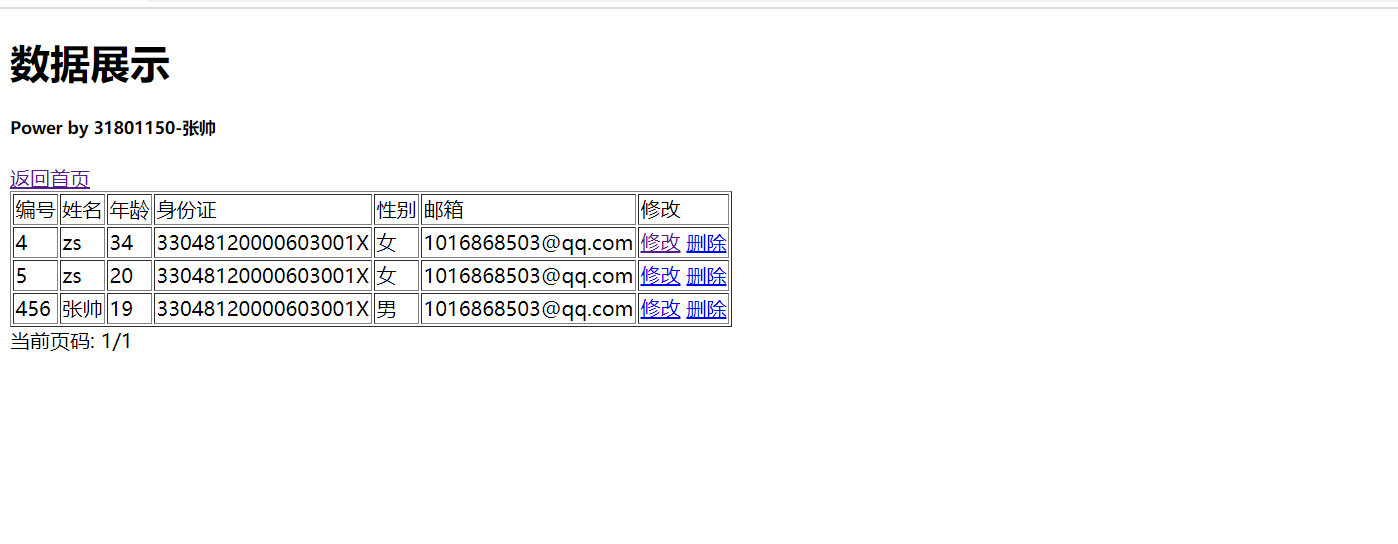
Mongodb界面：



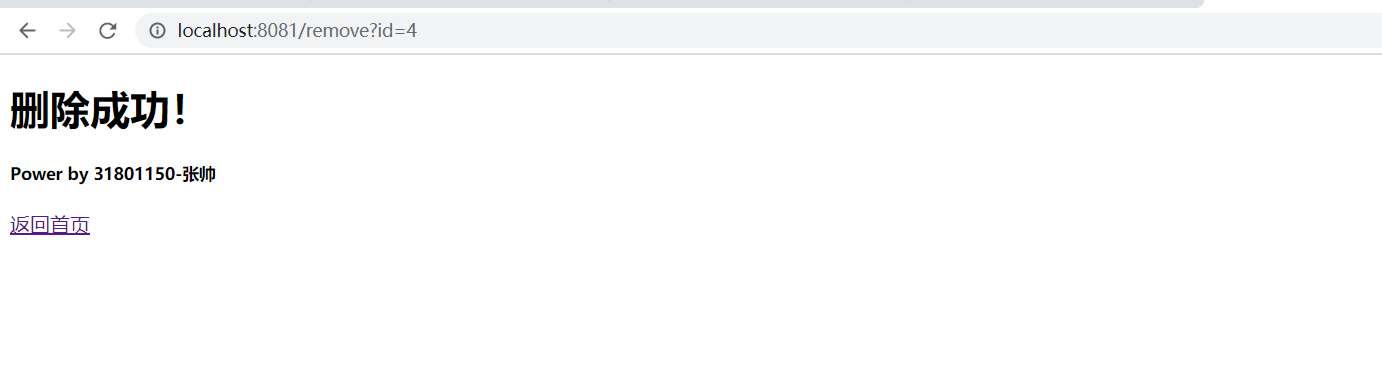
首页：



列表：



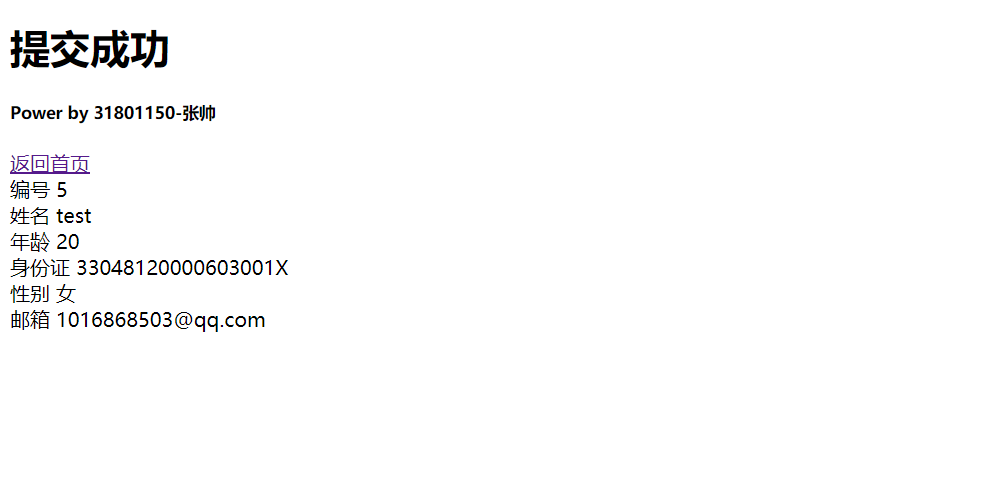
删除数据：

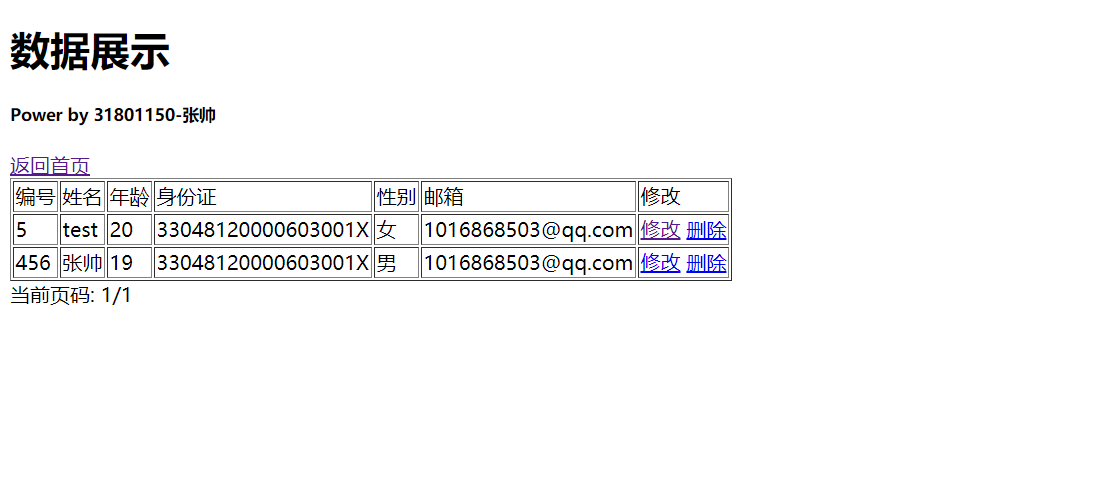




修改数据：







添加数据：



