**2022년도 졸업프로젝트**

**팀이름: 황제펭귄**

주제: TLS를 구현을 통한 홈페이지 제작 및 보안 채널 로그인 기능 구현

|  |  |  |  |
| --- | --- | --- | --- |
| **작성자** | 이유민 | **작성일자** | 2022.05.24 |

* **금일 진행 상황**

|  |  |
| --- | --- |
| **진행 상황** | **비고** |
| **진행 중**  **완료**   1. **거래 내역 조회 기능 생성**   **AccountService**  **- Transaction Data 저장 기능 추가**  // 전달자의 거래내역을 저장 transactionRepository.save(Transaction.*builder*()  .sendAccount(transferReq.getMyAccountNumber())  .receiveAccount(transferReq.getSendAccountNumber())  .balance(-transferReq.getBalance())  .build()); // 받는이의 거래내역을 저장 transactionRepository.save(Transaction.*builder*()  .sendAccount(transferReq.getSendAccountNumber())  .receiveAccount(transferReq.getMyAccountNumber())  .balance(transferReq.getBalance())  .build());  **TransactionException**  public class TransactionException extends RuntimeException{   public TransactionException(ExceptionMessages exceptionMessages) {  super(exceptionMessages.getMessage());  }   public TransactionException(String message){  super(message);  } }  **TransactionInquiryRes**  @Data @Builder @RequiredArgsConstructor @AllArgsConstructor public class TransactionInquiryRes {   private Long senderAccount;  private Long receiverAccount;  private Long sendMoney; }  **TraEventRequestRes**  @Data @Builder @RequiredArgsConstructor @AllArgsConstructor public class EventRequestRes {   private String title;  private String content;  private String startDate;  private String endDate; }  **Transaction(Entity)**  @Builder @Getter @Entity @NoArgsConstructor @AllArgsConstructor public class Transaction {   @Id  @GeneratedValue(strategy = GenerationType.*IDENTITY*)  private Long transactionId;   @Column  private Long sendAccount;   @Column  private Long receiveAccount;   @Column(nullable = false)  private Long balance;   public TransactionInquiryRes toDto() {  if (balance > 0) {  return TransactionInquiryRes.*builder*()  .senderAccount(receiveAccount)  .receiverAccount(sendAccount)  .sendMoney(balance)  .build();  } else {  return TransactionInquiryRes.*builder*()  .senderAccount(sendAccount)  .receiverAccount(receiveAccount)  .sendMoney(balance)  .build();  }  } }   1. **Event Test 추가**   @Builder @Getter @Entity @NoArgsConstructor @AllArgsConstructor public class Transaction {   @Id  @GeneratedValue(strategy = GenerationType.*IDENTITY*)  private Long transactionId;   @Column  private Long sendAccount;   @Column  private Long receiveAccount;   @Column(nullable = false)  private Long balance;   public TransactionInquiryRes toDto() {  if (balance > 0) {  return TransactionInquiryRes.*builder*()  .senderAccount(receiveAccount)  .receiverAccount(sendAccount)  .sendMoney(balance)  .build();  } else {  return TransactionInquiryRes.*builder*()  .senderAccount(sendAccount)  .receiverAccount(receiveAccount)  .sendMoney(balance)  .build();  }  } }  **진행 예정** |  |
| * **특이사항 / 협업 사항** | |