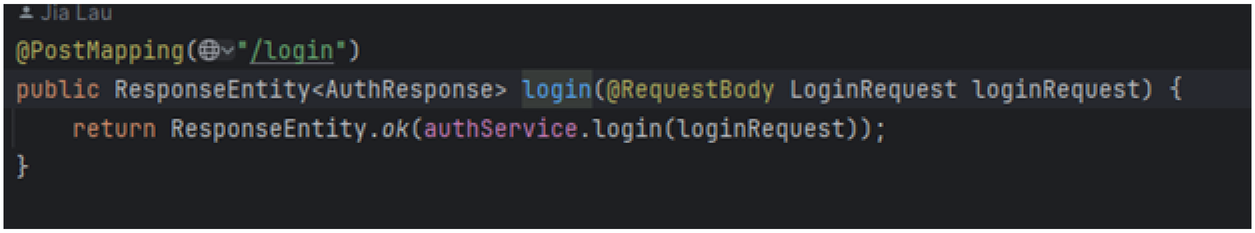
1. Authentication API

1.1. Login Authentication

Purpose: Allows users to login

Method: POST

Process: Checks if email is correct -> If Valid(will provide token) -> Returns AuthResponse which contains token.



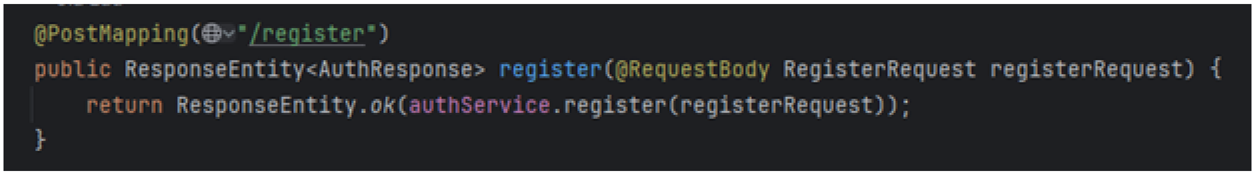
```
Jia Lau
@PostMapping("/login")
public ResponseEntity<AuthResponse> login(@RequestBody LoginRequest loginRequest) {
    return ResponseEntity.ok(authService.login(loginRequest));
}
```

1.2. Login Registration

Purpose: Allows users to register for an account

Method: POST

Process: Checks if user exist in database -> Creates a new user object -> Saves into database



```
@PostMapping("/register")
public ResponseEntity<AuthResponse> register(@RequestBody RegisterRequest registerRequest) {
    return ResponseEntity.ok(authService.register(registerRequest));
}
```

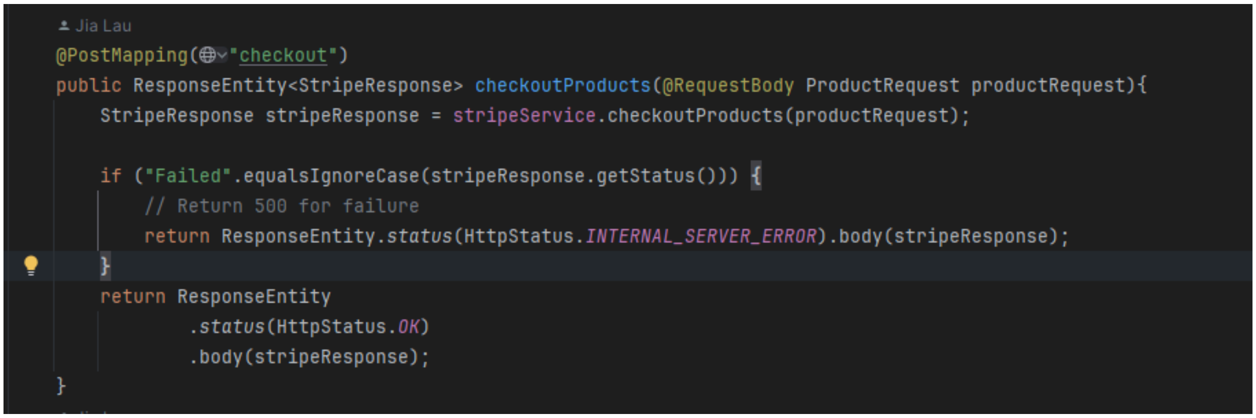
2. Stripe API

2.1. Checkout

Purpose: Allows users to purchase a singular item

Method: POST

Process: Takes in ProductRequest(Price_Id,Quantity) -> Sets payment session with stripe -> Success into stripe checkout page



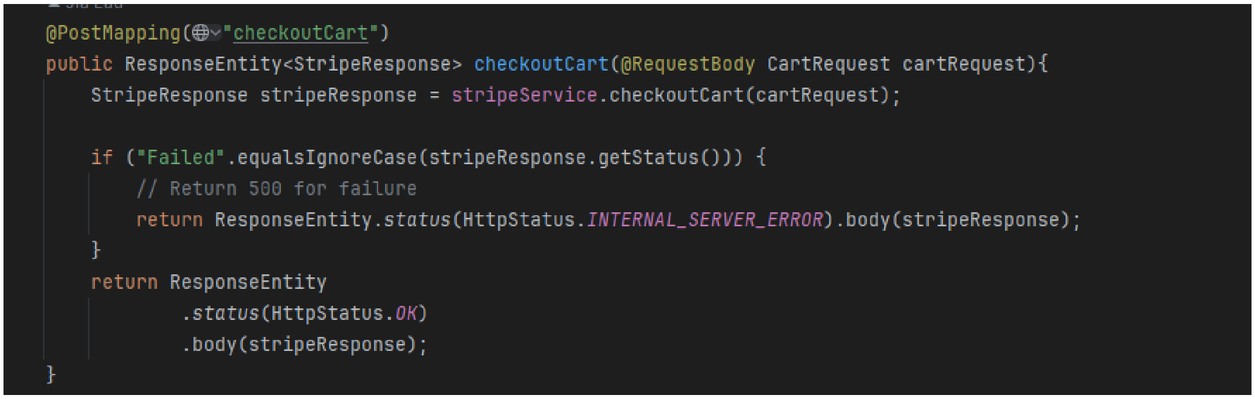
```java
@PostMapping("checkout")
public ResponseEntity<StripeResponse> checkoutProducts(@RequestBody ProductRequest productRequest){
    StripeResponse stripeResponse = stripeService.checkoutProducts(productRequest);

    if ("Failed".equalsIgnoreCase(stripeResponse.getStatus())) {
        // Return 500 for failure
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).body(stripeResponse);
    }
    return ResponseEntity
            .status(HttpStatus.OK)
            .body(stripeResponse);
}
```

2.2. CheckoutCart

Purpose: Allows users to purchase multiple items (Cart)

Method: POST

Process: Takes in CartRequest(List of ProductRequest) -> Sets payment session with stripe -> Success into stripe checkout page.



```java
@PostMapping("checkoutCart")
public ResponseEntity<StripeResponse> checkoutCart(@RequestBody CartRequest cartRequest){
    StripeResponse stripeResponse = stripeService.checkoutCart(cartRequest);

    if ("Failed".equalsIgnoreCase(stripeResponse.getStatus())) {
        // Return 500 for failure
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).body(stripeResponse);
    }
    return ResponseEntity
            .status(HttpStatus.OK)
            .body(stripeResponse);
}
```

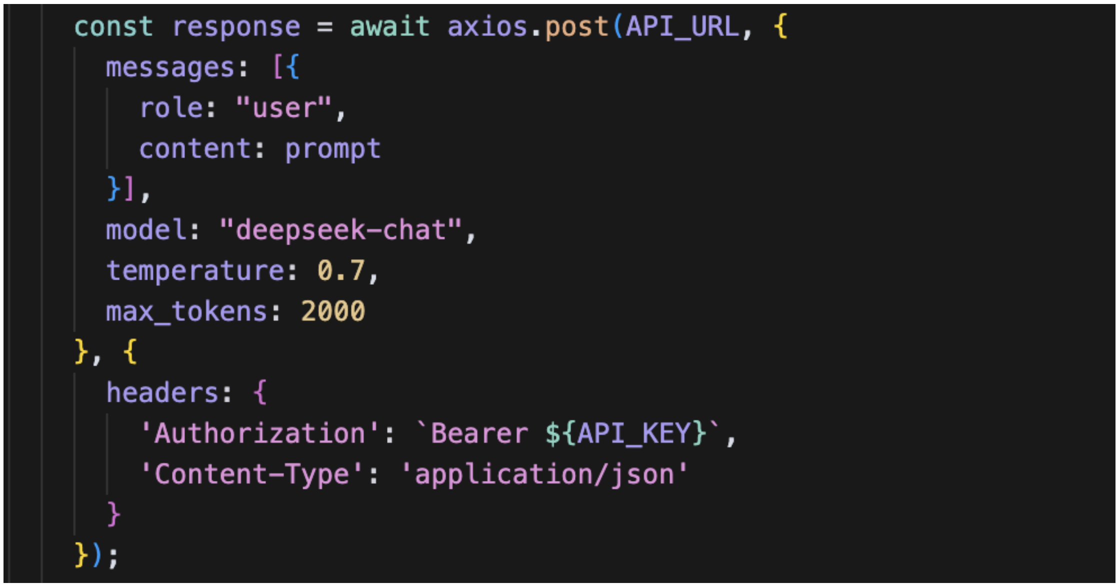
3. DeepSeek API

3.1 Workout prompt

Purpose: Generate Workout Routine

Method: POST

Process: Receives user fitness preferences, constructs AI prompt, sends to DeepSeek API, processes response, and returns structured workout plan with daily exercises, sets, reps, and training tips.



```
const response = await axios.post(API_URL, {
  messages: [{
    role: "user",
    content: prompt
  }],
  model: "deepseek-chat",
  temperature: 0.7,
  max_tokens: 2000
}, {
  headers: {
    'Authorization': `Bearer ${API_KEY}`,
    'Content-Type': 'application/json'
  }
});
```