Controller

package com.brokerportal.controller;

import com.brokerportal.entity.\*;

import com.brokerportal.service.\*;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

import java.util.Optional;

@RestController

@RequestMapping("/brokers")

public class BrokerController {

@Autowired

private BrokerService brokerService;

@GetMapping

public List<Broker> getAllBrokers() {

return brokerService.getAllBrokers();

}

@GetMapping("/{brokerId}")

public ResponseEntity<Broker> getBrokerById(@PathVariable Long brokerId) {

Optional<Broker> broker = brokerService.getBrokerById(brokerId);

return broker.map(ResponseEntity::ok).orElseGet(() -> ResponseEntity.notFound().build());

}

@PostMapping

public Broker saveBroker(@RequestBody Broker broker) {

return brokerService.saveBroker(broker);

}

}

@RestController

@RequestMapping("/clients")

public class ClientController {

@Autowired

private ClientService clientService;

@GetMapping

public List<Client> getAllClients() {

return clientService.getAllClients();

}

@GetMapping("/{clientId}")

public ResponseEntity<Client> getClientById(@PathVariable Long clientId) {

Optional<Client> client = clientService.getClientById(clientId);

return client.map(ResponseEntity::ok).orElseGet(() -> ResponseEntity.notFound().build());

}

@PostMapping

public Client saveClient(@RequestBody Client client) {

return clientService.saveClient(client);

}

}

@RestController

@RequestMapping("/locations")

public class LocationController {

@Autowired

private LocationService locationService;

@GetMapping("/{zipCode}")

public ResponseEntity<Location> getLocationByZipCode(@PathVariable String zipCode) {

Location location = locationService.getLocationByZipCode(zipCode);

return location != null ? ResponseEntity.ok(location) : ResponseEntity.notFound().build();

}

@PostMapping("/{zipCode}")

public Location fetchAndStoreLocation(@PathVariable String zipCode) {

return locationService.fetchAndStoreLocationDetails(zipCode);

}

}

@RestController

@RequestMapping("/quotes")

public class QuoteController {

@Autowired

private QuoteService quoteService;

@GetMapping

public List<Quote> getAllQuotes() {

return quoteService.getAllQuotes();

}

@GetMapping("/{quoteId}")

public ResponseEntity<Quote> getQuoteById(@PathVariable Long quoteId) {

Optional<Quote> quote = quoteService.getQuoteById(quoteId);

return quote.map(ResponseEntity::ok).orElseGet(() -> ResponseEntity.notFound().build());

}

@GetMapping("/client/{clientId}")

public List<Quote> getQuotesByClientId(@PathVariable Long clientId) {

return quoteService.getQuotesByClientId(clientId);

}

@PostMapping

public Quote createQuote(@RequestBody QuoteDTO quoteDTO) {

return quoteService.createQuote(quoteDTO);

}

@PutMapping("/{quoteId}")

public ResponseEntity<Quote> updateQuote(@PathVariable Long quoteId, @RequestBody QuoteDTO quoteDTO) {

return ResponseEntity.ok(quoteService.updateQuote(quoteId, quoteDTO));

}

@DeleteMapping("/{quoteId}")

public ResponseEntity<Void> deleteQuote(@PathVariable Long quoteId) {

quoteService.deleteQuote(quoteId);

return ResponseEntity.noContent().build();

}

}

@RestController

@RequestMapping("/premiums")

public class PremiumController {

@Autowired

private PremiumService premiumService;

@GetMapping

public List<Premium> getAllPremiums() {

return premiumService.getAllPremiums();

}

@PostMapping("/{quoteId}")

public ResponseEntity<Premium> savePremium(@PathVariable Long quoteId) {

return ResponseEntity.ok(premiumService.savePremium(quoteId));

}

}