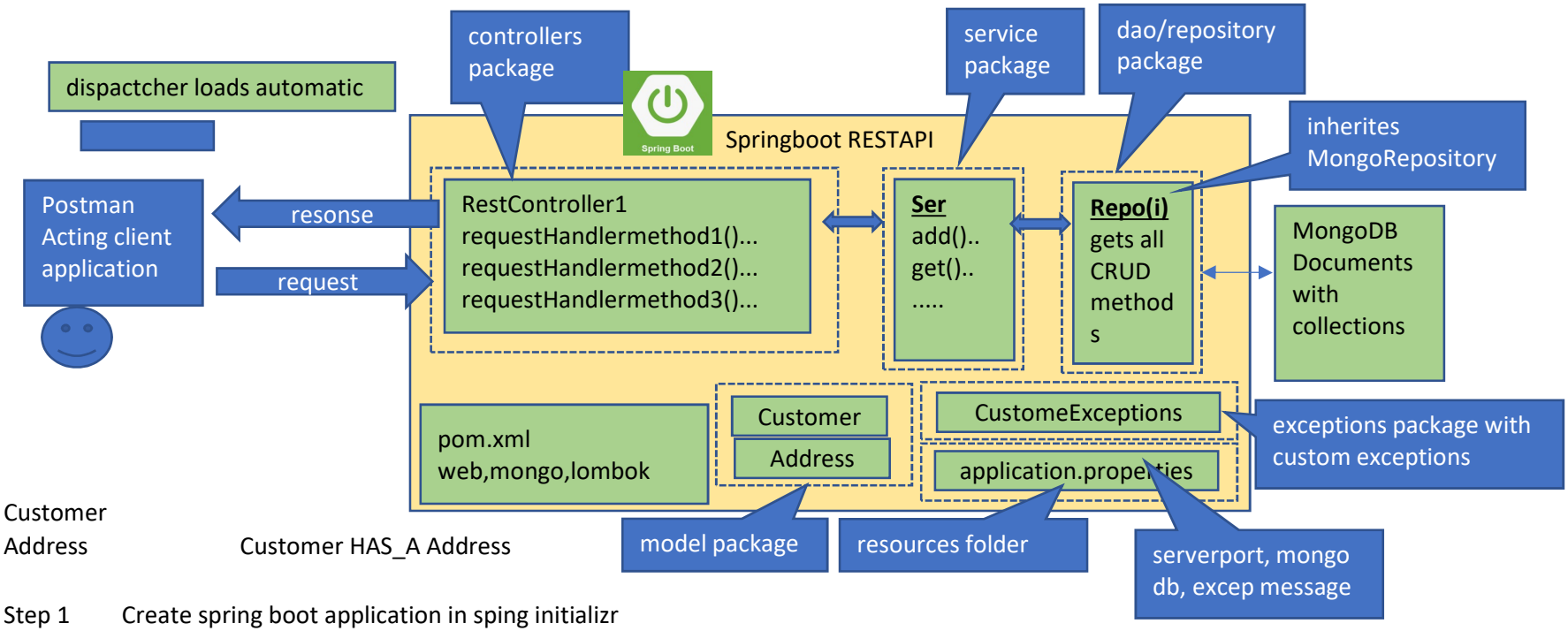
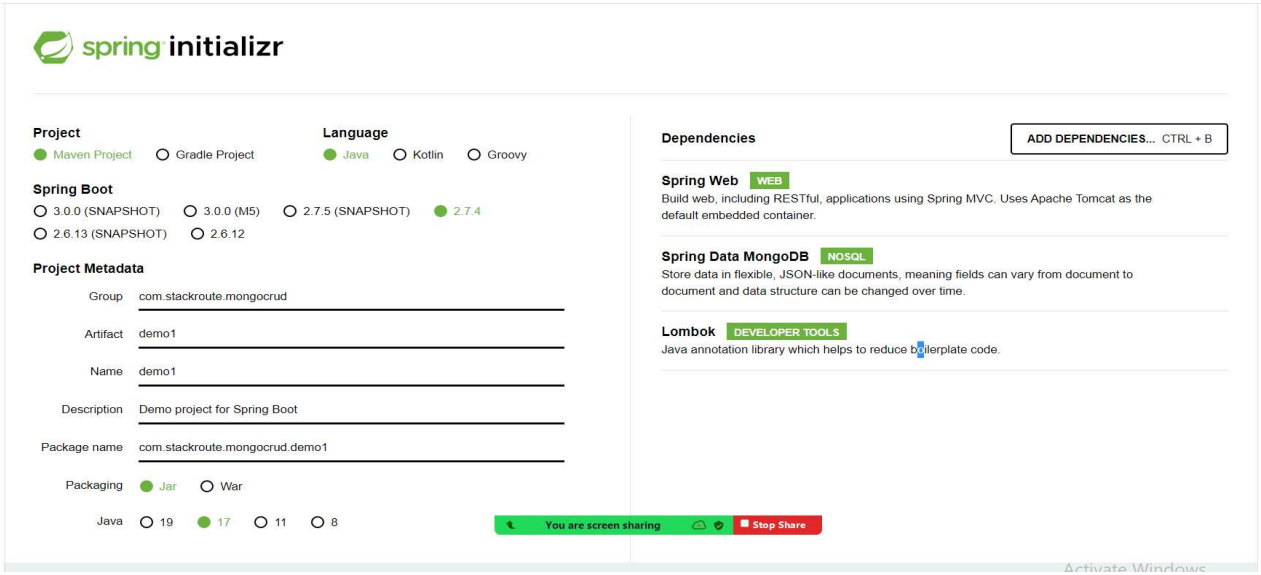
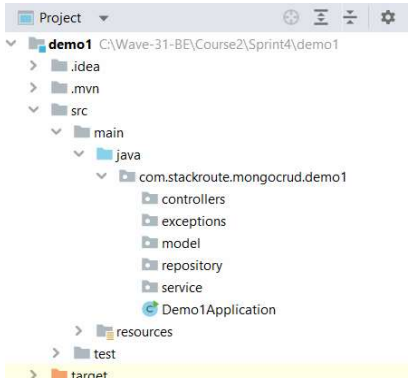


Customer
Address Customer HAS_A Address

Step 1 Create spring boot application in sping initializr



Step 2 download application, open in intellij, create required packages



Step 3 Define model classes

Customer HAS_A Address

```java
@NoArgsConstructor
@AllArgsConstructor
@Data
@Document
public class Customer {
    @Id
    private String customerId;
    private String name, email,mobile;
    private Address address;
}
```

```java
@NoArgsConstructor
@AllArgsConstructor
@Data
public class Address {
    private String doorNo, street, area, city;
}
```

Step 4 Create custom exceptions

```java
@ResponseStatus(code= HttpStatus.NOT_FOUND, reason="Customer not found")
public class CustomerNotFoundException extends Exception {
}
```

```java
@ResponseStatus(code= HttpStatus.CONFLICT, reason="Customer already exists")
public class CustomerAlreadyExistingException extends Exception{
}
```

Step 5 Create repository

```java
public interface CustomerRepository extends MongoRepository<Customer, String> {
}
```

Step 6 Create service layer

```java
public interface CustomerService {
    public abstract List<Customer> getAllCustomers();
    public abstract Customer addCustomer(Customer customer) throws CustomerAlreadyExistingException;
    public abstract Customer getCustomerById(String customerId) throws CustomerNotFoundException;
}
```

```java
    @Override
    public Customer addCustomer(Customer customer) throws CustomerAlreadyExistingException {
        // if new customer customerid existing in db, throw excep
        // else add record and return response
        if( customerRepository.findById(customer.getCustomerId()).isPresent()){
            throw new CustomerAlreadyExistingException();
        }
        else {
            return customerRepository.insert(customer);
        }
    }
```

```java
    @Override
    public Customer getCustomerById(String customerId) throws CustomerNotFoundException {
        //
        if(customerRepository.findById(customerId).isPresent()){
            return customerRepository.findById(customerId).get();
        }
        else{
            throw new CustomerNotFoundException();
        }
    }
```

Step 7 Define controller

```java
@RequestMapping("/customerapp/v1")
public class CustomerController {
    3 usages
    @Autowired
    private CustomerService customerService;

    @GetMapping("/customer")
    public ResponseEntity<?> getAllCustomers(){
        return new ResponseEntity<>(customerService.getAllCustomers(), HttpStatus.OK);
    }

    @PostMapping("/customer")
    public ResponseEntity<?> addCustomer(@RequestBody Customer customer) throws CustomerAlreadyExistingException{
        try{
            return new ResponseEntity<>(customerService.addCustomer(customer), HttpStatus.OK);
        }
        catch(CustomerAlreadyExistingException ex){
            throw new CustomerAlreadyExistingException();
        }
    }

    @GetMapping("/get-customer-by-id/{customerId}")
    public ResponseEntity<?> getCustomerById(@PathVariable String customerId) throws CustomerNotFoundException {
        try{
            return new ResponseEntity<>(customerService.getCustomerById(customerId),HttpStatus.OK);
        }
        catch(CustomerNotFoundException ex){
            throw new CustomerNotFoundException();
        }
    }
}
```


Step 8 edit application.properties

```
server.port=9999
spring.data.mongodb.database=sprint4
spring.data.mongodb.uri=mongodb://localhost:27017/
spring.error.include-message=always
```

customer query for getting customers by city

in repository

```
2 usages
public interface CustomerRepository extends MongoRepository<Customer, String> {

    1 usage
    @Query("{'address.city' : {$in:[?0]}}")
    public abstract List<Customer> getCustomersByCity(String city);
}
```

in service

```
public abstract List<Customer> getCustomersByCity(String city);
```

```
    @Override
    public List<Customer> getCustomersByCity(String city) {
        return customerRepository.getCustomersByCity(city);
    }
```

Controller

```
    // GET
    // http://localhost:9999/customerapp/v1/get-customers-by-city/Chennai
    @GetMapping("/get-customers-by-city/{city}")
    public ResponseEntity<?> getCustomersByCity(@PathVariable String city){
        return new ResponseEntity<>(customerService.getCustomersByCity(city), HttpStatus.OK);
    }
```