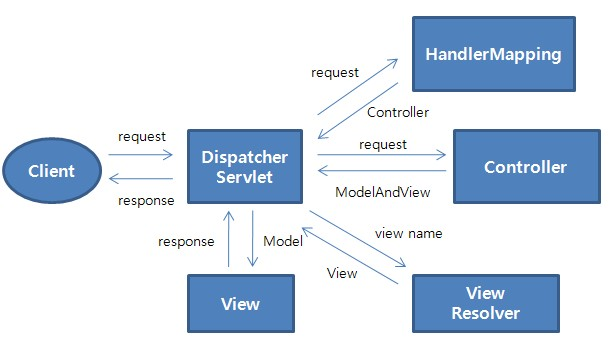
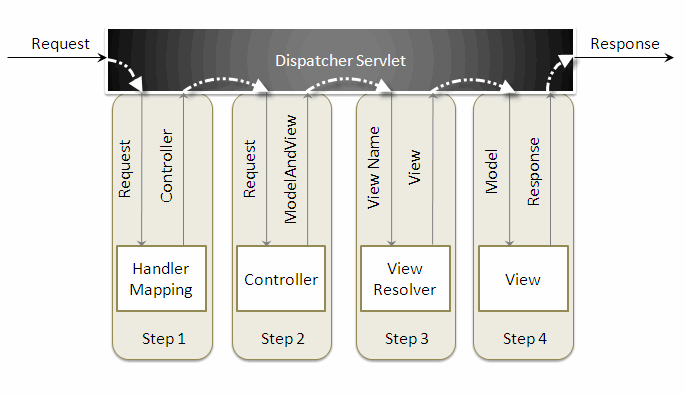
Quiz 11: Quiz

CS544 Enterprise Architecture

1. Explain what the Front Controller Pattern is:





* The *DispatcherServlet* first receives the request.
* The *DispatcherServlet* consults the *HandlerMapping* and invokes the *Controller* associated with the request.
* The *Controller* process the request by calling the appropriate service methods and returns a *ModeAndView* object to the *DispatcherServlet*. The *ModeAndView* object contains the model data and the view name.
* The *DispatcherServlet* sends the view name to a *ViewResolver* to find the actual *View* to invoke.
* Now the *DispatcherServlet* will pass the model object to the *View* to render the result.
* The *View* with the help of the model data will render the result back to the user.

1. Explain what a Flash Attribute is in the context of Spring MVC:

Attribute used for temporary storage with redirects. Stores it in the Session for just the duration of the redirect (next request). Remove after the first time page rendered.

1. Write the code for a Spring Data JPA interface called PizzaDao. This is not an open book or laptop quiz! Just do the best you can from memory.

@Repository

public interface PizzaDao extends JpaRepository <Pizza, Long>{}

1. Write the code for a Spring MVC controller class called PizzaController that has the PizzaDao injected and has the following methods: getAll(), viewAdd(), add(), get(), update(), delete()  
   Hint: these are the same methods as that were on the Car and Book controller in the homework.

@Controller

public class PizzaController {

@Autowired

private PizzaDao pizzaDao;

@GetMapping("/pizzas")

public String getAll(Model model) {

model.addAttribute("pizzas", pizzaDao.findAll());

return "pizzaList";

}

@GetMapping("/pizzas/add")

public String viewAdd(@ModelAttribute Pizza pizza, Model model) {

//model.addAttribute("pizza", new Pizza());

return "pizzaForm" //If you’re using Spring Form form

}

@PostMapping("/pizzas")

public String add(Pizza pizza) {

pizzaDao.save(pizza);

return "redirect:/pizzas";

}

@GetMapping("/pizzas/{id}")

public String get(@PathVariable long id, Model model) {

Optional<Pizza> pizza = pizzaDao.findById(id);

model.addAttribute("pizza", pizza.get());

return "pizzaDetail";

}

@PostMapping("/pizzas/{id}")

public String update(Pizza pizza) {

pizzaDao.save(pizza);

return "redirect:/pizzas";

}

@PostMapping("/pizzas/delete")

public String delete(@RequestParam long id) {

pizzaDao.deleteById(id);

return "redirect:/pizzas";

}

}