1. HTTP protocol

1. **Обяснете статус кодоветена response-ите, които започват с 1xx, 2xx, 3xx, 4xx и 5xx?**
   - **1xx** - Информативни кодове, които например може да се използват за смяна на проткол или например статус 100 continue, което дава на клиента информация, че може да продължи с операцията която прави. С тази група кодове, сървърът потвърждава заявката, инициирана от браузъра (клиента), и че тя се обработва;
   - **2xx** - Успешни кодове. Получена, разбрана, обработена и очаквана информация, предадена на браузъра;
   - **3xx** - Кодове за пренасочване. Поисканият ресурс е заменен с друга дестинация. Може да са необходими допълнителни действия от браузъра;
   - **4xx** - Клиентска грешка. Уебсайтът или страницата не са достигнати. Страницата е недостъпна или е имало технически проблем със заявката;
   - **5xx** - Сървърна грешка. Заявката е приетам но поради грешка на сървъра не се е изпълнила.

2. **HTTP/2 - какво е multiplexing и server push?**
   Multiplexing е възможността да се пратят паралелно много заявки по една единствена връзка (по една единствена TCP конекция).
   Server push - позволява на сървъра да връща проактивно (без да са поискани от потребителя) множество файлове наведнъж.

3. **Как работи TLS/SSL?**
   Това е криптиран канал от 6-то ниво (Presentation) на OSI модела. Обменя се ключ между клиента и сървъра, който се използва за криптиране на връзката, за да не може трето лице да се намеша по средата и да злоупотреби с информацията, която се предава.

4. **Какви HTTP глаголи/операции познавате?**
   GET - за извличане на ресурс;
   POST - за създаване на ресурс;
   PUT - за обновяване на ресурс;
   PATCH - за частично обновяване на ресурс;
   DELETE - за изтриване на ресурс;
   OPTIONS - за връщане на някаква информация, като например какви други методи поддържа приложението.

5. **Опишете структурата на едно URL.**
   Унифициран локатор на ресурси (URL) – стандартизиран адрес на даден мрежов ресурс (документ или страница)

   http://en.wikipedia.org:80/w/index.php?search=Students&dd=true#last

   схема, FQDN или хост, порт, път, низ на заявката, фрагмент идентификатор (anchor)

6. **Каква е разликата между URI и URL?**
   URI - Uniform Resource Identifier; **URL** - Uniform Resource Locator.
   URL-ите са подмножество на URI-те и съдържат в себе си механизъм за вземане на ресурс.

7. **Каква е раликата между PUT/POST/PATCH HTTP методите?**
   PUT - цялостно обновяване, POST - създаване и PATCH - частитна актуализация на ресурс.

8. **Какво представлява моделът POST-redirect-GET и защо се изпллзва?**
   Когато направим POST заявка за създаване на някакъв ресурс, той се създава на сървъра и обикновено не е много удачно да върнем директно response-а, тъй като

при refresh на страницата - браузърът ще се опита отново да създаде ресурса. За това е удачно да върнем статус REDIRECT и да пренасочим към URL-а на новия request и браузърът да направи GET заявка.

2. Spring Boot Introduction

1. **Какво представляват Beans в Spring?**
   Това са различни обекти, между които общото е, че се менажират от Spring. Те живеят в така наречения Application Context. Spring ги инициализира, менажира и унищожава. Целия жизнен цикъл на тези обекти се управлява от Spring. Казано с други думи - не използваме оператора new когато искаме да създадем даден Bean, а вместо това, за да получим достъп до тях изпозлваме IoC и DI (Inversion of Control & Dependency Injection).

2. **За какво служи @PostConstruct анотацията?**
   @PostConstruct анотация се слага върху метод, който трябва да бъде извикан след като е инициализиран един Spring Bean.

3. **За какво служи PreDestroy анотацията?**
   PreDestroy анотация се слага върху метод, които се извиква от контейнера, когато се унищожава някаквъ Bean.

4. **Има ли разлика между предназначението на application.properties и application.yml файловете.**
   Не. И двата файла са с еднакво предназначение, но структурата на информацията в тях е различна. Форматът в yml или yaml фаиловете е по-удобен и по-четим за хора, поради йерархичната си структура, която също така и редуцира текста във файла, но разчита на отстъпа (indent-а) на редовете.

5. **Какви Bean scopes познавате? Дайте примери и посочете какви са разликите между тях.**
   Основните Bean scope-ове са "singleton" и "prototype". Singleton се инициализира един път и играе ролята на singleton в application context-а. Prototype се създава всеки път когато има нужда се използва. Другите scope-ове са "request", "session", "websocket" и други custom scope-ове.

6. **Какво е Spring Boot?**
   Spring Boot е начин, по който може да си конфигурираме нашето Spring приложение с помощтта на удобен dependency management, auto configuration и т.н. Както и името му подсказва - да си BOOTstrap-нем нашето Spring приложение.

7. **Само на Java ли се пишат Spring приложения?**
   Не. Пишат се и на други JVM-based езици като Kotlin, Scala, Groovy, Clojure, Fantom. Ceylon, Jython.

3. Spring MVC

1. **Как анотираме контролер класовете?**
   Анотираме ги с @Controller анотацията, а когато имплементираме REST API използваме @RestController, което е @Controller + @ResponseBody.

2. **Как използваме @RequestMapping анотацията?**
   Използваме я на ниво клас и ни дава префикс на всички методи в класа на контролера. @RequestMapping може да се използва и на ниво метод, като обикновено в такъв случай се задава освен пътя и метода (GET, POST, etc.).

3. **Има ли разлики между @GetMapping и @RequestMapping с метод GET върху метод?**
   Не, няма. Едно и също нещо е. @GetMapping е alias път към @RequestMapping с

метод GET.
Следните са еквивалентни:
@RequestMapping(value = "/ex/foo", headers = "key=val", method = GET)
и
@GetMapping(value = "/ex/foo", headers = "key=val")

4. **Как може да вземем POST параметър?**
   С помощта на DTO binding model и с @RequestParam анотацията.

5. **Как предаваме обект към Thymeleaf view?**
   С помощта на обектите, които Spring ни позволява да инжектираме в методите на контролерите. Това са Model, ModelMap и ModelAndView.

6. **Как може да вземем optional параметър?**
   В @RequestParam анотацията има параметър required, който е с булева стойност и приема true или false.

7. **Как може да зададем (default) стойност по подразбиране на параметър?**
   В @RequestParam анотацията има параметър defaultValue, който приема дефолтната стойност като String.

8. **Каква е разликата между Path variable и Query parameter?**
   Path variable е променлива, която се кодира в пътя на URL-а (localhost:8080/orders/3/details), а Query parameter-а е променлива, която се кодира в query string-а (localhost:8080/orders?id=3)

4. State Management

1. **Какво е Cookie?**
   Cookie-то е малък текстов файл, който се създава на сървъра и се изпраща от сървъра към клиента (например уеб браузъра). Този текстов файл не съдържа изпълним код, а ключ-стойност параметри.

2. **За какво служат Cookies?**
   Служат за персонализация, за user tracking и най-често се използват за session management. В Cookie се държи SESSION-ID-то, което отговаря на HTTP сесията, която се пази на сървъра.

3. **Какво е Third Party Cookie?**
   Third party cookie-тата се използват за user tracking и за реклами. Когато отвориме сайт А и там се изпълнява скрипт от сайт В и той сложи свое cookie там. Т.е. отваряме сайт А, а всъщност се появява cookie от сайт В на нашия браузър. Когато отвориме сайт С, отново се зарежда cookie-то от сайт В и сайт С може да прочете cookie-то на база на него да препоръча нещо специфично (сайт С ще знае къде и какво сме предпочитали, което е в разрез с нарушаване на личните данни и за това тези third party cookies ще се използват до края на 2023).

4. **Какво е HTTP сесия?**
   Това е една област в паметта на сървъра (или в базата данни, до която има достъп сървъра), където се пази информация за даден user. Може да слагаме най-различни неща в сесията, но тя обикновенно е свързана с някакъв идентификационен номер, който се предава между клиента и сървъра, като най-често това става с cookie.

5. **Как може в Sring да добавим атрибут в HTTP Session?**
   Като инжектираме HttpSession обекта като аргумент и го използваме за да му добавим атрибут с метода му addAttribute.

6. **Как може да се четат Cookies в контролер?**
   Чрез @CookieValue анотацията.

7. **Как може сървърът да изтрие едно Cookie?**
   Като сетне същото cookie със същото име, със същите пропъртита но със max-age=0, което ще доведе до изтриването на cookie-то.

5. Spring Essentials

1. **Какво е Thymeleaf?**
   Thymeleaf е модерна сървърна машина за шаблони на Java както за уеб, така и за самостоятелни среди.
   Основната цел на Thymeleaf е да внесе елегантни шаблони в процеса на разработка - HTML, който може да се показва правилно в браузърите и също така да работи като статични прототипи, което позволява по-силно сътрудничество в екипите за разработка.
   С модули за Spring Framework, множество интеграции с любимите ви инструменти и възможност за включване на ваша собствена функционалност, Thymeleaf е идеален за съвременната HTML5 JVM уеб разработка – въпреки че може да направи много повече.

2. **Какво представляват Thymeleaf фрагментите?**
   HTML фрагменти за повторно използване на някои общи части на сайт.

3. **Защо използваме link expressions?**
   За да свържем адрес с даден HTML препратка. Релативен път се свързва с контекста на нашия уеб сървър. Можем лесно да навигираме през нашите шаблонни файлове. Абсолютен път се свърза със посочения ресурс дори и извън нашия сървър.

4. **Обяснете за какво служи @Primary анотацията?**
   @Primary за задаване на приоритет на Bean, когато има множество Bean-ове от един и същ тип. Когато Bean е анотиран с @Primary, той ще се инстанцира, ако не е зададен уточняваш @Qualifier на инстанцирания обект от еквивалентент тип.

5. **Какво е Same Origin и CORS?**
   Политиката за същия произход е критичен механизъм за сигурност, който ограничава как документ или скрипт, зареден от един източник, може да взаимодейства с ресурс от друг произход.
   CORS е механизъм за сигурност, който позволява на уеб страница от един домейн или Origin да има достъп до ресурс с различен домейн (заявка между домейни). CORS е облекчаване на политиката за същия произход, внедрена в съвременните браузъри.

6. **С коя анотация управляваме CORS?**
   @CrossOrigin

7. **Дайте пример за употреба на CORS.**
   ```
   @CrossOrigin
   @GetMapping("/{id}")
   public Account retrieve(@PathVariable Long id) {
       ...
   }
   ```

8. **За какво служи @ModelAttribute анотацията върху метод?**
   Създава Bean на ниво контролер, който може да се достъпва в Thymeleaf.