

in 1
1 out of
question

Кои од следните искази се точни?

Select one or more:

- Кога во рамки на користењето на EntityGraph, type = EntityGraph.EntityGraphType.LOAD, атрибутите кои се специфицирани преку јазлите на граffот ќе бидат третирани како FetchType.LAZY, а останатите како FetchType.EAGER
- Default fetch type за @ManyToOne релација е FetchType.EAGER
- Default fetch type за @OneToMany релација е FetchType.LAZY
- Кога креирааме проекција истото мора да го направиме користејќи класа.
- Кога во рамки на користењето на EntityGraph, type = EntityGraph.EntityGraphType.LOAD, атрибутите кои се специфицирани преку јазлите на граffот ќе бидат третирани како FetchType.EAGER, а останатите зависно нивниот специфициран или default FetchType

in 2
2 out of
question

Кои од следните искази се точни?

Select one or more:

- Ако додадеме својство **@Lock private Long versionLock** во рамки на одреден ентитет овозможуваме имплементација на Optimistic Locking.
- Кога има потреба од користење на материјализиран поглед, истиот мора да биде обновен во одредени ситуации. Ваквото обновување може да се реализира преку метод во persistence (repository) слојот, кој се повикува во сервисната логика.
- 0 15 10 * * ? 2010** е cron израз кој означува извршување секој ден во 15:10 часот за време на годината 2010.
- 0 0 12 * * ?** е cron израз кој означува извршување секој ден во 12 часот на пладне.
- 0 15 10 ? * *** е cron израз кој означува извршување секој ден во 15:10 часот.
- За да имплементираме scheduled task секогаш мора да користиме cron изрази.
- Ако додадеме својство **@Version private Long version** во рамки на одреден ентитет овозможуваме имплементација на Optimistic Locking.
- Optimistic locking е единствениот тип на locking кој постои.

3
1 out of
question

Избери која од понудените опции дава валидна синтакса во ReactJS за прикажување на детали за предмет во една студиска програма. Името на компонентата е **Course**. Податоците за курсот се предаваат како својство на следниот начин:

```
let obj={  
    id:1, courseName:"EMT", courseFund:"2+2+1"  
};  
  
<course course={obj}/>
```

Select one:

import React, {Component} from 'react'

```
class Course extends Component {  
    render() {  
        return (  
            <li key="{this.props.course.id}" className="list-group-item">  
                <div>  
                    <div className="row">  
                        <div className="col-md-2">  
                            {this.course.id}  
                        </div>  
                        <div className="col-md-6">  
                            {this.course.courseName}  
                        </div>  
                        <div className="col-md-4">  
                            {this.course.courseFund}  
                        </div>  
                    </div>  
                </div>  
            </li>  
        )  
    }  
    export default Course
```

```
○ import React, {Component} from 'react'

class Course extends Component {
  constructor(props) {
    super(props);
  }
  render() {
    return (
      <li key="{this.props.course.id}" className="list-group-item">
        <div>
          <div className="row">
            <div className="col-md-2">
              {this.props.course.id}
            </div>
            <div className="col-md-6">
              {this.props.course.courseName}
            </div>
            <div className="col-md-4">
              {this.props.course.courseFund}
            </div>
          </div>
        </div>
      </li>
    )
  }
}

export default Course
```

```
○ import React, {Component} from 'react'

class Course extends Component {
  return() {
    render(
      <li className="list-group-item">
        <div>
          <div className="row">
            <div className="col-md-2">
              {props.course.id}
            </div>
            <div className="col-md-6">
              {props.course.courseName}
            </div>
            <div className="col-md-4">
              {props.course.courseFund}
            </div>
          </div>
        </div>
      </li>
    )
  }
}
export default Course
```

```
○ import React, {Component} from 'react'

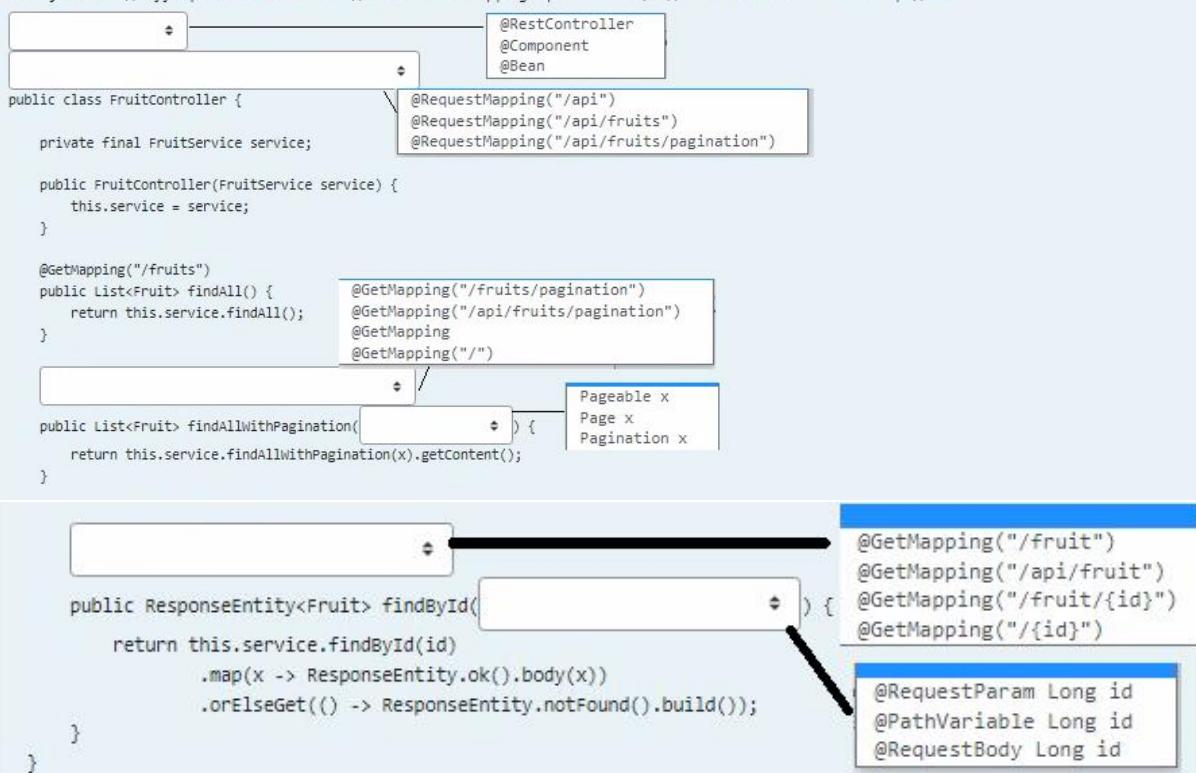
class Course extends Component {
  render() {
    return (
      <li className="list-group-item">
        <div>
          <div className="row">
            <div className="col-md-2">
              {this.props.id}
            </div>
            <div className="col-md-6">
              {this.props.courseName}
            </div>
            <div className="col-md-4">
              {this.props.courseFund}
            </div>
          </div>
        </div>
      </li>
    )
  }
}
export default Course
```

Нека е даден следниот код за **FruitController**. Потребно е истиот да се дополни за да се обезбеди:

findAll() метод кој ги враќа сите овошки на mapping `/api/fruits`

findAllWithPagination() метод кој ги враќа сите овошки на mapping `/api/fruits/pagination`

findById() метод кој ги враќа овошката со соодветното id на mapping `/api/fruit?id=ID`, каде ID е заменет со вистинска вредност



5

Што од понуденото побарува ДДД дизајнот:

out of
question

- Select one or more:
- a. Користење на строг монолитен развој на апликацијата.
 - b. Страга дефиниција на кориснички интерфејс кој ќе биде лесно разбиралив од корисниците.
 - c. Грубо дефинирање на проблемот со алатките од тактички дизајн, и потоа нивно полирање со алатките од стратегиски дизајн.
 - d. Стратегијата за именување на ентитетите и операциите со нив е строго именувана од доменските експерти.
 - e. Тим кој вклучува софтверски инженери и експерти од доменот при градење на софтверското решение.
 - f. Поделба на доменските модели во помали кластери со користење на стратегиски шаблон наречен ограничен контексти (Bounded Contexts).
 - g. Грубо дефинирање на проблемот со алатките од стратегиски дизајн, и потоа нивно полирање со алатките од тактички дизајн.
 - h. Дефинирање на јазик кој ќе овозможува ефикасна тимска комуникација наречен Сеопфатен јазик (Ubiquitous Language).

6

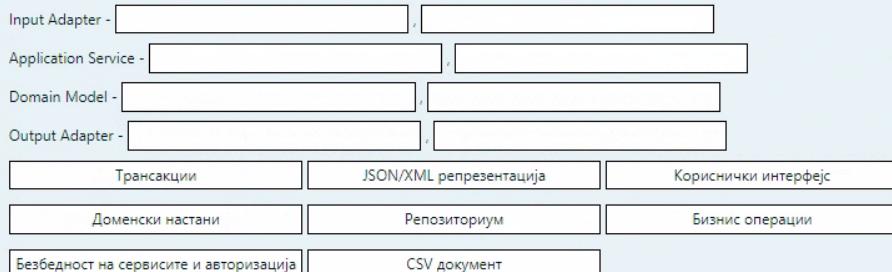
Кои од понудените искази се точни во врска со Агрегати (Aggregates), Корен на агрегат (Aggregate Root)?

out of
question

- Select one or more:
- a. Промените кои треба да се рефлектираат во другите агрегати од страна на еден агрегат, треба да бидат постојано синхронизирани и во рамки на истата трансакција да се изврши нивната модификација.
 - b. Доменски настан (Domain Event) се практикат само преку коренот на агрегатот.
 - c. Секој агрегат се состои од повеќе ентитети, каде што само еден ентитет се нарекува Корен на агрегатот.
 - d. Операциите кои предизвикуваат промена на состојбата се извршуваат во рамки на секој ентитет во агрегатот.
 - e. Еден агрегат може да е референциран од надворешниот свет само преку Коренот на агрегатот.
 - f. Секоја промена на локалните ентитети во еден агрегат е независна и коренот на агрегатот нема потреба да е свесен за истата.
 - g. Трансакцискиот интегритет секогаш се одржува во рамки на агрегатот.
 - h. Состојбата на агрегатите задолжително мора да е постојано конзистентна и коренот на агрегатот е одговорен за спроведување на сите бизнис правила во рамки на агрегатот.

7
d
out of
question

Асоцирајте ги соодветните компоненти со соодветниот слој од архитектурата на еден ограничен контекст (bounded context):



8
d
out of
question

Кои од следните изрази се точни за ентитети (Entities) и вредносни објекти (Value Objects) во ДДД?

Select one or more:

- a. Проверката дали два ентитета се исти се прави само со споредба на нивните примарни клучеви.
- b. Ентитетите немаат историја.
- c. Проверката дали два вредносни објекти се исти се прави само со споредба на нивните примарни клучеви.
- d. Сите атрибути на ентитетите задолжително е да поседуваат setter методи.
- e. Вредносните објекти треба да имаат неменлива природа.
- f. Сите атрибути во ентитетите не треба да се менливи (immutable) по нивната иницијализација.
- g. Вредносните објекти немаат историја.
- h. Сите атрибути на ентитетот не се менуваат со едноставни setter методи поради потребата за зачувување на промената во логот на настани.

1
d
out of
question

Која од следните особини не е карактеристична за електронската трговија?

- a. Социјална технологија
- b. Персонализација/прилагодување
- c. Предводено од масовен маркетинг (Mass-marketing driven)
- d. Обилност на информации (information richness)

2
d
out of
question

Која од наведените карактеристики не носи предност над конкурентите:

- a. поевтини добавувачи.
- b. поквалитетни производи.
- c. помалку производи.
- d. повеши работени.

3
d
out of
question

Кој дел од бизнис моделот описува како фирмата ќе заработка, ќе креира профит и ќе овозможи "супериорно" враќање на вложениот капитал

- a. предност пред конкуренцијата (competitive advantage)
- b. понудена вредност (value proposition)
- c. стратегија за пазарот (market strategy)
- d. модел на заработка (revenue model)

14

id
out of
question

Кои од компаниите/платформите заработкаат од претплата (subscription)?

- Tmall
- ReCAPTCHA
- Spotify
- Taobao
- Groupon
- AdWords
- Alibaba
- Dollar Shave Club
- Aliexpress
- Tinder
- PetSmart

in 6

ted
1 out of

Објаснете ги сличносите и разликите помеѓу вирус (Virus) и црев (Warm).

5

id
out of
question

Описете го користењето, дадете пример и наведете барем по две предности за ваше решение (**in-house**) за софтверот, и надворешно решение (**outsourcing**) за хардверот при градење на сајт за е-трговија.

Образложете кое решение (**in-house/outsourcing**) за софтвер и хардвер би било најсоодветно за стартап компанија.

1. Кои од понудените концепти припаѓаат на ДДД дизајнот?
 - ▷ Data Access Object (DAO)
 - ▷ Data Transfer Object (DTO)
 - ▶ **слоевита архитектуа**
 - ▷ Model View – View Model
 - ▶ **агрегати**
 - ▶ **Ограничени контексти (Bounded Contexts)**
 - ▷ model view controller (MVC)
2. Избери која од понудените опции дава валидна синтакса во ReactJS за прикажување на детали за испит по одреден предмет. Името на компонентата е Exam. Податоците за курсот се предаваат како својство на следниот начин:

```
let obj = { id:1, examName:"EMT 1" , courseName:"EMT"}  
<Exam exam = {obj}/>
```

Select one:

- ▷ import React, {Class} from 'react'

```
class Exam extends Class{  
  
    return() {  
        <li className="list-group-item">  
            <div className="row">  
                <div className="col-md-2">  
                    {this.props.exam.id}  
                </div>  
                <div className="col-md-6">  
                    {this.props.exam.examName}  
                </div>  
                <div className="col-md-4">  
                    {this.props.exam.courseName}  
                </div>  
            </div>  
        </li>  
  
    }  
}  
export default Exam
```



```

        {exam.examName}
    </div>
    <div className="col-md-4">
        {exam.courseName}
    </div>
</div>
</li>
)
}

}
export default Exam

```

3. Што од понуденото е точно за доменските настани (Domain Events)?
 - ▷ Доменските настани може да се публикуваат преку презентацијскиот слој на апликацијата.
 - ▷ Не содржат идентификатор, тој е карактеристичен само за ентитетите.
 - ▷ Домен настаните може по потреба да чуваат комплексна бизнис логика.
 - ▶ **Аплицирањето на промените во агрегатот и зачувувањето на доменскиот настан секогаш се прават во рамки на една трансакција.**
 - ▶ **Содржат временска марка за креирањето на настанот.**
 - ▷ Нивната состојба не е конзистентна и нивните атрибути може често да бидат променети по нивното креирање.
 - ▶ **Доменските настани се корисни доколку постои механизам на нивна надежна достава до оние кои се препратиле на истите.**

4. Кои од следните изрази се точни за ентитети (Entities) и вредносни објекти (Value Objects) во ДДД?
 - ▷ Вредносни објекти живеат самостојно и не зависат од останатите ентитети.
 - ▶ **Еднаквост меѓу два вредностни објекти од иста класа се утврдува врз база на еднаквоста на сите атрибути кои ги поседуваат.**
 - ▶ **За ентитетите се чува историска информација за промени.**
 - ▷ Сите атрибути во ентитетите не треба да се менливи (Immutable) по нивната иницијализација.
 - ▷ Вредносните објекти не смее да содржат бизнис логика, туку задолжително содржат само атрибути.
 - ▶ **Еднаквоста меѓу два ентитета од иста класа се утврдува врз база на примарен клуч.**

- ▶ Вредносните објекти не живеат самостојно, туку најчесто припаѓаат на некој ентитет.
 - ▶ Во вредносните објекти се чува комплексна бизнис логика и се оркестира менацирањето на ентитетите.
5. Што од следново е точно за JWT токените?
- ▶ **JWT е кратенка од JSON Web Token**
 - ▶ **Сетирањето на безсостојбена сесија во Spring Security се прави со следната конфигурација:**
`sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS)`
 - ▶ **Најчесто се корисни при авторизација на корисници, но и за безбедна размена на информации.**
 - ▶ **Содржината на заглавјето кога се пристапува заштитен ресурс треба да го користи следниот формат: Authorization: Bearer <token>**
 - ▶ Се состојат од три дела: заглавје (header), приватен клуч (private key) и потпис (signature)
 - ▶ **JWT токените се најчест избор кога сакаме да користиме безсостојбена автентификација.**
 - ▶ Најчесто се корисни при авторизација на корисници, но се лош избор при потреба од безбедна размена на информации
6. Нека е даден следниот код за FruitController. Потребно е истиот да се дополни за да се обезбеди:
- `findAll()` метод кој ги враќа сите овошки на mapping /api/fruits
`findAllWithPagination()` метод кој ги враќа сите овошки на mapping /api/fruits/pagination
`findById()` метод кој ја враќа овощката со соодветно id на mapping /api/fruit?id=ID, каде ID е заменет со вистинска вредност
- ```
@RestController
@RequestMapping("/api")

public class FruitController{
 private final FruitService service;
 public FruitController(FruitService service){
 this.service=service;
 }

 @GetMapping("fruits")
 public List<Fruit> findAll() {
 return this.service.findAll();
```

```

 }

 @GetMapping("/fruits/pagination")
 public List<Fruit> findAllWithPagination(Pageable x) {
 return this.service.findAllWithPagination(x).getContent();
 }

 @GetMapping("/fruit")
 public ResponseEntity<Fruit> findById(@RequestParam Long id) {
 return this.service.findById(id)
 .map(x->ResponseEntity.ok().body(x))
 .orElseGet(() -> ResponseEntity.notfound().build());
 }
}

```

7. Кои од следните искази се точни?

- ▷ За да имплементираме scheduled task потребно е секогаш да искористиме cron израз како овој 0 0 12 \* \* ?
- ▷ Ако додадеме својство @Lock private Long VersionLock во рамки на одреден ентитет овозможуваме имплементација на Optimistic Locking
- ▶ **Кога има потреба од користење на материјализиран поглед, истиот мора да биде обновен во одредени ситуации. Ваквото обновување може да се реализира преку метод во persistence repository слојот, кој се повикува во сервисната логика**
- ▶ **Ако додадеме својство @Version private long version во рамки на одреден ентитет овозможуваме имплементација на Optimistic Locking**

8. Кои од следните искази се точни?

- ▷ Кога во рамки на користењето EntityGraph.type= EntityGraph.EntityGraphType.LOAD, атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.EAGER, а останатите како FetchType.LAZY
- ▷ Default fetch type за @ManyToMany релација е FetchType.EAGER
- ▶ **Default fetch type за @ManyToOne релација е FetchType.EAGER**
- ▶ **Кога во рамки на користењето EntityGraph.type= EntityGraph.EntityGraphType.FETCH, атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.EAGER, а останатите како FetchType.LAZY**

9. Изберете што е точно од понуденото земајќи го предвид следното сценарио:

Потребно е да изработите систем кој прави пресметки на цената на автомобил врз база на цена на деловите од кои е составен. Дополнително, се води сметка и за бројот на

автомобилски делови кои стојат на расположување во магацин. Ентитетите кои се користат во сценариото се следните: Car, CarPart и Warehouse.

- ▷ Бизнис логиката за пресметката на цената на автомобилот треба да биде поствена во CarPart ентитетот.
- ▷ Ентитетот CarPart е корен на сопствениот агрегат
- ▶ **Бизнис логиката за пресметка на цената на автомобилот треба да биде поствена во Car ентитетот.**
- ▶ **Во сценариото постојат два агрегати, еден ги содржи ентитетите Car и CarPart, другиот го содржи само ентитетот Warehouse.**
- ▷ **Ентитетот Car е корен на сопствениот агрегат.**
- ▷ Постојат три агрегати, посебен за секој ентитет. Секој од ентитетите е сопствен корен на агрегатот.

10. Која од понудените опции нуди валидна синтакса за креирање на едноставна компонента GreetingComponent за приказ на следниот HTML код:

```
<div>
 Welcome to the party!
</div>
```

- ▷ import React, {Directive} from 'react'  
export default GreetingComponent extends Directive{  
 render(){  
 return(  
 <div>  
 {"<span> Welcome to the party! </span>"}  
 </div>  
 )  
 }  
}
- ▷ import React, {Class} from 'react'  
class GreetingComponent extends Class {  
 render(){  
 let message =<span> Welcome to the party! </span>  
 return(  
 <div></div>  
 {message}  
 )  
 }  
}

- ▷ 

```
import React, {Component} from 'react'
class GreetingComponent extends Component {
 render(){
 const message = Welcome to the party!
 return(
 <div>
 {message}
 </div>
)
 }
 export default GreetingComponent
```
  
- ```
import React from 'react';
const GreetingComponent = (props) => {
  let msg =<span> Welcome to the party! </span>
  return(
    <div>
      {msg}
    </div>
  )
}
export default GreetingComponent
```

11. Која од понудените компоненти користи валидна синтакса за креирање на едноставна компонента GoodbyeComponent за приказ на следниот HTML код:

```
<div>
```

```
  <b>Goodbye!</b>
```

```
</div>
```

- ```
import React from 'react';
const GoodbyeComponent = (props) => {
 let msg =Goodbye!
 return(
 <div>
 {msg}
 </div>
)
}
```

```
)
}

export default goodbyeComponent

▷ import React from 'react'
class GoodbyeComponent extends React.Component {
 render(){
 const message =Goodbye!

 return(
 <div>
 {message}
 </div>
)
 }
export default goodbyeComponent

▷ import React, {Class} from 'react'
class GoodbyeComponent extends Class {
 render(){
 const message =Goodbye!

 return(

 {message}

)
 }
export default goodbyeComponent
```

```

 ▷ import React from 'react'
 const goodbyeComponent = () => {
 let message =Goodbye!

 render(
 <div>
 {{msg}}
 </div>
)
 }
 export default goodbyeComponent

```

12. Кои од понудените искази се точни во врска со Агрегати (Aggregates), Корен на агрегат (Aggregate Root), Доменски настани (Domain Events)?

- ▶ Промените кои треба да се рефлектираат во другите агрегати од страна на еден агрегат се прави преку концептот на **Eventual consistency**.
- ▶ Состојбата на агрегатите задолжително мора да е постојано конзистентна и коренот на агрегатите е одговорен за спроведување на сите бизнис правила во рамки на агрегатот.
- ▶ Еден агрегат може да има повеќе корени (roots).
- ▶ Трансакцискиот интегритет секогаш се оддржува во рамки на цел ограничен контекст (Bounded Context).
- ▶ **Доменските настани може да се пратат преку коренот на агрегатот и доменските сервиси.**
- ▶ Промените кои треба да се рефлектираат во другите агрегати, од страна на еден агрегат, треба да бидат постојано синхронизирани и во рамки на истата трансакција да се извржи нивната модификација
- ▶ **Еден агрегат може да е референциран од надворешниот свет само преку коренот на агрегатот.**
- ▶ Секоја промена на локалните ентитети во еден агрегат е независна и коренот на агрегатот нема потреба да е свесен за истата.

13. Што од понуденото е дел од ДДД дизајнот:

- ▷ Кориснички интерфејс (User Interface)
- ▷ Искуство при користење (User Experience)
- ▷ Сеопфатен јазик (**Ubiquitous Language**)
- ▷ Ограничени контексти (**Bounded Contexts**)
- ▷ **Доменски настани (Domain Events)**

- ▷ Микросервисна архитектура
- ▷ Стратегија за маркетинг (Marketing Strategy)
- ▶ **Агрегати (Aggregates)**

14. Нека е даден следниот код за StudentController. Потребно е истиот да се дополни за да се обезбеди:

**findAll()** метод кој ги враќа сите студенти на mapping /api/students

**findAllWithPagination()** метод кој ги враќа сите студенти на mapping/api/students/pages

**findByName()** метод кој ја враќа студентот со соодветно име на mapping /api/student/NAME, каде NAME е заменет со вистинска вредност

```
@RestController
@RequestMapping("/api")
```

```
public class StudentController {
 private final StudentService service;
 public StudentController(StudentService service){
 this.service=service;
 }

 @GetMapping("/students")
 public List<Student> findAll() {
 return this.service.findAll();
 }

 @GetMapping("/students/pages")
 public List<Student> findAllWithPagination(Pageable x) {
 return this.service.findAllWithPagination(x).getContent();
 }

 @GetMapping("/student/{name}")
 public ResponseEntity<Student> findByName(@PathVariable String name) {
 return this.service.findByName(name)
 .map(x->ResponseEntity.ok().body(x))
 .orElseGet(() -> ResponseEntity.notfound().build());
 }
}
```

15. Кои од следните искази се точни?

- ▷ Default fetch type за @ManyToMany релација е FetchType.EAGER
- ▶ **Default fetch type за @ManyToOne релација е FetchType.EAGER**
- ▶ Кога во рамки на користењето EntityGraph.type= EntityGraph.EntityGraphType.LOAD, атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.LOAD, а останатите како FetchType.EAGER
- ▶ **Кога во рамки на користењето EntityGraph.type=EntityGraph.EntityGraphType.LOAD, атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.EAGER, а останатите зависно нивниот специфициран или default FetchType**

16. Кои од следните изрази се точни за ентитети (Entities) и вредносни објекти (Value Objects) во ДДД?

- ▷ Сите атрибути во ентитетите не треба да се менливи (Immutable) по нивната иницијализација.
- ▶ **Сите атрибути во вредносните објекти треба да бидат неменливи после иницијализација.**
- ▷ Во вредносните објекти се чува комплексната бизнис логика и се оркестрира менацирањето на ентитетите.
- ▶ **Вредносните објекти немаат примарен клуч.**
- ▷ Ентитетите и вредносните објекти ја чуваат својата историја на менување во база.
- ▶ **Вредносните објекти немаат историја.**
- ▷ Ентитетите немаат историја.
- ▶ **Ентитетите имаат примарен клуч.**

17. Што од следново е точно за JWT токените?

- ▶ **JWT е кратенка од JSON Web Token**
- ▶ Се состојат од три дела: заглавје (header), податоци (payload) и потпис (signature)
- ▶ Најчесто се корисни при авторизација на корисници, но и за безбедна размена на информации.
- ▶ Содржината на заглавјето кога се пристапува заштитен ресурс треба да го користи следниот формат: Authorization: Bearer <token>.
- ▶ **JWT токените се најчест избор кога сакаме да користиме безсостојбена автентификација.**
- ▷ Содржината на заглавјето кога се пристапува заштитен ресурс треба да го користи следниот формат: Bearer: Authorization <token>.
- ▷ JWT токените не поддржуваат безсостојбена автентификација.
- ▷ Најчесто се корисни при авторизација на корисници, но се лош избор при безбедна размена на информации.

18. Кога најчесто се креира Anticorruption слој?

- ▶ Кога е потребно да се овозможи изолација помеѓу два зависни домени кои имаат сопствен сеопфатен јазик (**Ubiquitous Language**).
- ▶ Кога има потреба од слој кој прави превод помеѓу два различни домена кои користат сопствени сеопфатни јазици (**Ubiquitous Languages**).
- ▷ Кога постои јасно дефиниран протокол кој треба да се имплементира за да се направи интеграција помеѓу два домена.
- ▷ Кога преводот меѓу два сеопфатни јазици (**Ubiquitous Languages**) е невозможен и има потреба од директна интеграција на моделот на едниот домен во другиот

1. Кои од понудените концепти припаѓаат на ДДД дизајнот?
  - ▷ Data Access Object (DAO)
  - ▷ Data Transfer Object (DTO)
  - ▶ **слоевита архитектуа**
  - ▷ Model View – View Model
  - ▶ **агрегати**
  - ▶ **Ограничени контексти (Bounded Contexts)**
  - ▷ model view controller (MVC)
2. Избери која од понудените опции дава валидна синтакса во ReactJS за прикажување на детали за испит по одреден предмет. Името на компонентата е Exam. Податоците за курсот се предаваат како својство на следниот начин:

```
let obj = { id:1, examName:"EMT 1" , courseName:"EMT"}
<Exam exam = {obj}/>
```

Select one:

- ▷ import React, {Class} from 'react'

```
class Exam extends Class{

 return() {
 <li className="list-group-item">
 <div className="row">
 <div className="col-md-2">
 {this.props.exam.id}
 </div>
 <div className="col-md-6">
 {this.props.exam.examName}
 </div>
 <div className="col-md-4">
 {this.props.exam.courseName}
 </div>
 </div>

 }
}
export default Exam
```



```

 {exam.examName}
 </div>
 <div className="col-md-4">
 {exam.courseName}
 </div>
</div>

)
}

}
export default Exam

```

3. Што од понуденото е точно за доменските настани (Domain Events)?
  - ▷ Доменските настани може да се публикуваат преку презентацијскиот слој на апликацијата.
  - ▷ Не содржат идентификатор, тој е карактеристичен само за ентитетите.
  - ▷ Домен настаните може по потреба да чуваат комплексна бизнис логика.
  - ▶ **Аплицирањето на промените во агрегатот и зачувувањето на доменскиот настан секогаш се прават во рамки на една трансакција.**
  - ▶ **Содржат временска марка за креирањето на настанот.**
  - ▷ Нивната состојба не е конзистентна и нивните атрибути може често да бидат променети по нивното креирање.
  - ▶ **Доменските настани се корисни доколку постои механизам на нивна надежна достава до оние кои се препратиле на истите.**
  
4. Кои од следните изрази се точни за ентитети (Entities) и вредносни објекти (Value Objects) во ДДД?
  - ▷ Вредносни објекти живеат самостојно и не зависат од останатите ентитети.
  - ▶ **Еднаквост меѓу два вредностни објекти од иста класа се утврдува врз база на еднаквоста на сите атрибути кои ги поседуваат.**
  - ▶ **За ентитетите се чува историска информација за промени.**
  - ▷ Сите атрибути во ентитетите не треба да се менливи (Immutable) по нивната иницијализација.
  - ▷ Вредносните објекти не смее да содржат бизнис логика, туку задолжително содржат само атрибути.
  - ▶ **Еднаквоста меѓу два ентитета од иста класа се утврдува врз база на примарен клуч.**

- ▶ Вредносните објекти не живеат самостојно, туку најчесто припаѓаат на некој ентитет.
  - ▶ Во вредносните објекти се чува комплексна бизнис логика и се оркестира менацирањето на ентитетите.
5. Што од следново е точно за JWT токените?
- ▶ **JWT е кратенка од JSON Web Token**
  - ▶ **Сетирањето на безсостојбена сесија во Spring Security се прави со следната конфигурација:**  
`sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS)`
  - ▶ **Најчесто се корисни при авторизација на корисници, но и за безбедна размена на информации.**
  - ▶ **Содржината на заглавјето кога се пристапува заштитен ресурс треба да го користи следниот формат: Authorization: Bearer <token>**
  - ▶ Се состојат од три дела: заглавје (header), приватен клуч (private key) и потпис (signature)
  - ▶ **JWT токените се најчест избор кога сакаме да користиме безсостојбена автентификација.**
  - ▶ Најчесто се корисни при авторизација на корисници, но се лош избор при потреба од безбедна размена на информации
6. Нека е даден следниот код за FruitController. Потребно е истиот да се дополни за да се обезбеди:
- `findAll()` метод кој ги враќа сите овошки на mapping /api/fruits  
`findAllWithPagination()` метод кој ги враќа сите овошки на mapping /api/fruits/pagination  
`findById()` метод кој ја враќа овощката со соодветно id на mapping /api/fruit?id=ID, каде ID е заменет со вистинска вредност
- ```
@RestController
@RequestMapping("/api")

public class FruitController{
    private final FruitService service;
    public FruitController(FruitService service){
        this.service=service;
    }

    @GetMapping("fruits")
    public List<Fruit> findAll() {
        return this.service.findAll();
```

```

    }

    @GetMapping("/fruits/pagination")
    public List<Fruit> findAllWithPagination( Pageable x ) {
        return this.service.findAllWithPagination(x).getContent();
    }

    @GetMapping("/fruit")
    public ResponseEntity<Fruit> findById( @RequestParam Long id ) {
        return this.service.findById(id)
            .map(x->ResponseEntity.ok().body(x))
            .orElseGet(() -> ResponseEntity.notfound().build());
    }
}

```

7. Кои од следните искази се точни?

- ▷ За да имплементираме scheduled task потребно е секогаш да искористиме cron израз како овој 0 0 12 * * ?
- ▷ Ако додадеме својство @Lock private Long VersionLock во рамки на одреден ентитет овозможуваме имплементација на Optimistic Locking
- ▶ **Кога има потреба од користење на материјализиран поглед, истиот мора да биде обновен во одредени ситуации. Ваквото обновување може да се реализира преку метод во persistence repository слојот, кој се повикува во сервисната логика**
- ▶ **Ако додадеме својство @Version private long version во рамки на одреден ентитет овозможуваме имплементација на Optimistic Locking**

8. Кои од следните искази се точни?

- ▷ Кога во рамки на користењето EntityGraph.type= EntityGraph.EntityGraphType.LOAD, атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.EAGER, а останатите како FetchType.LAZY
- ▷ Default fetch type за @ManyToMany релација е FetchType.EAGER
- ▶ **Default fetch type за @ManyToOne релација е FetchType.EAGER**
- ▶ **Кога во рамки на користењето EntityGraph.type= EntityGraph.EntityGraphType.FETCH, атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.EAGER, а останатите како FetchType.LAZY**

9. Изберете што е точно од понуденото земајќи го предвид следното сценарио:

Потребно е да изработите систем кој прави пресметки на цената на автомобил врз база на цена на деловите од кои е составен. Дополнително, се води сметка и за бројот на

автомобилски делови кои стојат на расположување во магацин. Ентитетите кои се користат во сценариото се следните: Car, CarPart и Warehouse.

- ▷ Бизнис логиката за пресметката на цената на автомобилот треба да биде поствена во CarPart ентитетот.
- ▷ Ентитетот CarPart е корен на сопствениот агрегат
- ▶ **Бизнис логиката за пресметка на цената на автомобилот треба да биде поствена во Car ентитетот.**
- ▶ **Во сценариото постојат два агрегати, еден ги содржи ентитетите Car и CarPart, другиот го содржи само ентитетот Warehouse.**
- ▷ **Ентитетот Car е корен на сопствениот агрегат.**
- ▷ Постојат три агрегати, посебен за секој ентитет. Секој од ентитетите е сопствен корен на агрегатот.

10. Која од понудените опции нуди валидна синтакса за креирање на едноставна компонента GreetingComponent за приказ на следниот HTML код:

```
<div>
    <span>Welcome to the party!</span>
</div>
```

- ▷ import React, {Directive} from 'react'
export default GreetingComponent extends Directive{
 render(){
 return(
 <div>
 {" Welcome to the party! "}
 </div>
)
 }
}
- ▷ import React, {Class} from 'react'
class GreetingComponent extends Class {
 render(){
 let message = Welcome to the party!
 return(
 <div></div>
 {message}
)
 }
}

- ▷

```
import React, {Component} from 'react'
class GreetingComponent extends Component {
  render(){
    const message =<span> Welcome to the party! </span>
    return(
      <div>
        {message}
      </div>
    )
  }
  export default GreetingComponent
```

- ```
import React from 'react';
const GreetingComponent = (props) => {
 let msg = Welcome to the party!
 return(
 <div>
 {msg}
 </div>
)
}
export default GreetingComponent
```

11. Која од понудените компоненти користи валидна синтакса за креирање на едноставна компонента GoodbyeComponent за приказ на следниот HTML код:

```
<div>
```

```
 Goodbye!
```

```
</div>
```

- ```
import React from 'react';
const GoodbyeComponent = (props) => {
  let msg =<b>Goodbye!</b>
  return(
    <div>
      {msg}
    </div>
  )
}
```

```
)  
}  
  
export default goodbyeComponent  
  
▷ import React from 'react'  
class GoodbyeComponent extends React.Component {  
    render(){  
        const message =<b>Goodbye!</b>  
  
        return(  
            <div>  
                {message}  
            </div>  
        )  
    }  
export default goodbyeComponent  
  
▷ import React, {Class} from 'react'  
class GoodbyeComponent extends Class {  
    render(){  
        const message =<b>Goodbye!</b>  
  
        return(  
            <span>  
                {message}  
            </span>  
        )  
    }  
export default goodbyeComponent
```

```

    ▷ import React from 'react'
      const goodbyeComponent = () => {
        let message =<b>Goodbye!</b>

        render(
          <div>
            {{msg}}
          </div>
        )
      }
      export default goodbyeComponent

```

12. Кои од понудените искази се точни во врска со Агрегати (Aggregates), Корен на агрегат (Aggregate Root), Доменски настани (Domain Events)?

- ▶ Промените кои треба да се рефлектираат во другите агрегати од страна на еден агрегат се прави преку концептот на **Eventual consistency**.
- ▶ Состојбата на агрегатите задолжително мора да е постојано конзистентна и коренот на агрегатите е одговорен за спроведување на сите бизнис правила во рамки на агрегатот.
- ▶ Еден агрегат може да има повеќе корени (roots).
- ▶ Трансакцискиот интегритет секогаш се оддржува во рамки на цел ограничен контекст (Bounded Context).
- ▶ **Доменските настани може да се пратат преку коренот на агрегатот и доменските сервиси.**
- ▶ Промените кои треба да се рефлектираат во другите агрегати, од страна на еден агрегат, треба да бидат постојано синхронизирани и во рамки на истата трансакција да се извржи нивната модификација
- ▶ **Еден агрегат може да е референциран од надворешниот свет само преку коренот на агрегатот.**
- ▶ Секоја промена на локалните ентитети во еден агрегат е независна и коренот на агрегатот нема потреба да е свесен за истата.

13. Што од понуденото е дел од ДДД дизајнот:

- ▷ Кориснички интерфејс (User Interface)
- ▷ Искуство при користење (User Experience)
- ▷ Сеопфатен јазик (**Ubiquitous Language**)
- ▷ Ограничени контексти (**Bounded Contexts**)
- ▷ **Доменски настани (Domain Events)**

- ▷ Микросервисна архитектура
- ▷ Стратегија за маркетинг (Marketing Strategy)
- ▶ **Агрегати (Aggregates)**

14. Нека е даден следниот код за StudentController. Потребно е истиот да се дополни за да се обезбеди:

findAll() метод кој ги враќа сите студенти на mapping /api/students

findAllWithPagination() метод кој ги враќа сите студенти на mapping/api/students/pages

findByName() метод кој ја враќа студентот со соодветно име на mapping /api/student/NAME, каде NAME е заменет со вистинска вредност

```
@RestController
@RequestMapping("/api")
```

```
public class StudentController {
    private final StudentService service;
    public StudentController(StudentService service){
        this.service=service;
    }

    @GetMapping("/students")
    public List<Student> findAll() {
        return this.service.findAll();
    }

    @GetMapping("/students/pages")
    public List<Student> findAllWithPagination( Pageable x ) {
        return this.service.findAllWithPagination(x).getContent();
    }

    @GetMapping("/student/{name}")
    public ResponseEntity<Student> findByName( @PathVariable String name ) {
        return this.service.findByName(name)
            .map(x->ResponseEntity.ok().body(x))
            .orElseGet(() -> ResponseEntity.notfound().build());
    }
}
```

15. Кои од следните искази се точни?

- ▷ Default fetch type за @ManyToMany релација е FetchType.EAGER
- ▶ **Default fetch type за @ManyToOne релација е FetchType.EAGER**
- ▶ Кога во рамки на користењето EntityGraph.type= EntityGraph.EntityGraphType.LOAD, атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.LOAD, а останатите како FetchType.EAGER
- ▶ **Кога во рамки на користењето EntityGraph.type=EntityGraph.EntityGraphType.LOAD, атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.EAGER, а останатите зависно нивниот специфициран или default FetchType**

16. Кои од следните изрази се точни за ентитети (Entities) и вредносни објекти (Value Objects) во ДДД?

- ▷ Сите атрибути во ентитетите не треба да се менливи (Immutable) по нивната иницијализација.
- ▶ **Сите атрибути во вредносните објекти треба да бидат неменливи после иницијализација.**
- ▷ Во вредносните објекти се чува комплексната бизнис логика и се оркестрира менацирањето на ентитетите.
- ▶ **Вредносните објекти немаат примарен клуч.**
- ▷ Ентитетите и вредносните објекти ја чуваат својата историја на менување во база.
- ▶ **Вредносните објекти немаат историја.**
- ▷ Ентитетите немаат историја.
- ▶ **Ентитетите имаат примарен клуч.**

17. Што од следново е точно за JWT токените?

- ▶ **JWT е кратенка од JSON Web Token**
- ▶ Се состојат од три дела: заглавје (header), податоци (payload) и потпис (signature)
- ▶ Најчесто се корисни при авторизација на корисници, но и за безбедна размена на информации.
- ▶ Содржината на заглавјето кога се пристапува заштитен ресурс треба да го користи следниот формат: Authorization: Bearer <token>.
- ▶ **JWT токените се најчест избор кога сакаме да користиме безсостојбена автентификација.**
- ▷ Содржината на заглавјето кога се пристапува заштитен ресурс треба да го користи следниот формат: Bearer: Authorization <token>.
- ▷ JWT токените не поддржуваат безсостојбена автентификација.
- ▷ Најчесто се корисни при авторизација на корисници, но се лош избор при безбедна размена на информации.

18. Кога најчесто се креира Anticorruption слој?

- ▶ Кога е потребно да се овозможи изолација помеѓу два зависни домени кои имаат сопствен сеопфатен јазик (**Ubiquitous Language**).
- ▶ Кога има потреба од слој кој прави превод помеѓу два различни домена кои користат сопствени сеопфатни јазици (**Ubiquitous Languages**).
- ▷ Кога постои јасно дефиниран протокол кој треба да се имплементира за да се направи интеграција помеѓу два домена.
- ▷ Кога преводот меѓу два сеопфатни јазици (**Ubiquitous Languages**) е невозможен и има потреба од директна интеграција на моделот на едниот домен во другиот

1. Опишете што се Drive by downloads преку пример и објаснете како може една организација да се заштити од нив?

Кога компјутерот ќе се зарази со злонамерен софтвер едноставно со посета на веб-страница, тој е познат како „drive by download“. Овој тип на напад го добил името „drive by download“, бидејќи корисникот не мора да застанува или да кликнува било каде на злонамерната страница. Едноставно гледање на страницата е доволно за да предизвика инфекција што се случува во позадина и без знаење или согласност на корисникот.

Во drive by download напад, криминалците компромитираат веб-страница, честопати легитимна, со вградување или инјектирање малициозни објекти во внатрешноста на веб-страниците. Инфекциите се невидливи за корисникот и се движат од малициозен код на JavaScript до iFrames, врски, пренасочувања, малверзации, скриптирање меѓу страници и други малициозни елементи.

Кога корисникот посетува заразена веб -страница, прелистувачот на корисникот автоматски го вчитува малициозниот код, кој веднаш го скенира компјутерот на жртвата за безбедносни слабости во оперативниот систем и други апликации.

Како да се заштитиме од таков тип на вируси:

- Update your software quickly and constantly
- Remove unnecessary software and plug-ins
- Stop using a privileged account for day-to-day work.
- Use a firewall
- Disable Java and JavaScript
- Use web-filtering software
- Install an ad blocker

1. Кои од понудение искази се точни во врска со Агрегати (Aggregates), Корен на агрегат (Aggregate Root)?

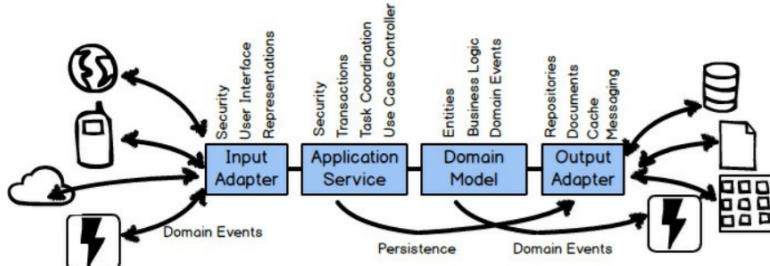
- a. Промените кои треба да се рефлектираат во другите агрегати од страна на еден агрегат, треба да бидат постојано синхронизирани и во рамки на истата трансакција да се изврши нивната модификација.
- b. Трансакцискиот интегритет секогаш се одржува во рамки на агрегатот
- c. Секој агрегат се состои од повеќе ентитети, каде што само еден ентитет се нарекува Корен на агрегатот
- d. Доменски настан (Domain Event) се праќаат само преку коренот на агрегатот.
- e. Операциите кои предизвикуваат промена на состојбата се извршуваат во рамки на секој ентитет во агрегатот.
- f. Секоја промена на локалните ентити во еден агрегат е независна и коренот на агрегатот нема потреба да е свесен за истата
- g. Состојбата на агрегатите задолжително мора да е постојано конзистентна и коренот на агрегатот е одговорен за спроведување на сите бизнис правила во рамки на агрегатот.
- h. Еден агрегат може да е референциран од надворешниот свет само преку коренот на агрегатот.

3. Одберете кои од понудените искази се точни во рамки на архитектурата на еден ограничен контекст.

- Трансакциите се менаџираат во Domain Service
- Безбедноста на надворешниот свет се менаџира во Output Adapter.
- Безбедноста на надворешниот свет се менаџира во Input Adapter.
- Трансакциите треба да ги менаџираат на ниво на Repository
- Безбедноста во бизнис операциите се менаџира во Input Adapter
- Трансакциите се менаџираат во Application Service
- Безбедноста во бизнис операциите се менажира во Domain Model
-

Architecture

There is another question that you may have wondered about. What's inside a *Bounded Context*? Using this *Ports and Adapters* [IDDD] architecture diagram, you can see that a *Bounded Context* is composed of more than a domain model.



These layers are common in a *Bounded Context*: *Input Adapters*, such as user interface controllers, REST endpoints, and message listeners; *Application Services* that orchestrate use cases and manage transactions; the domain model that we've been focusing on; and *Output Adapters* such as persistence management and message senders. There is much to be said about the various layers in this architecture, and it is too elaborate to state in this distilled book. See [Chapter 4 of Implementing Domain-Driven Design](#) [IDDD] for an exhaustive discussion.

Technology-Free Domain Model

Although there will be technology scattered throughout your architecture, the domain model should be free of technology. For one thing, that's why transactions are managed by the application services and not by the domain model.

4. Што од понуденото е дел од ДДД дизајнот:

- Агрегати (Aggregates)** не сум сигурна
- Ограничени контексти (Bounded Contexts)** точно
- Доменски настани (Domain Events)** не сум сигурна
- Стратегија за маркетинг (Marketing Strategy)
- Сеопфатен јазик (Ubiquitous Language)** точно
- Искуство при користење (User Experience)
- Кориснички интерфејс (User Interface)
- Микросервисна архитектура

5. Наведете пример за вашата апликација и направете груба процена за буџетот потребен за е присуство, земајќи ги предвид најчестите ставки кои влагаат во буџетот на веб-сајтовите. При процената, земете во предвид дека и за софтверот и за хардверот ќе користите надворешно решение (outsourcing). При тоа, образложете кои се погодностите на ваквите решенија за софтверот и хардверот во однос на спротивните опции (пр. outsourcing решение во однос на In-house решение)

6. Кои од следните изрази се точни за ентитети (Entities) и вредносни објекти (Value Objects) во ДДД?

- Вредносните објекти не смее да содржат бизнис логика, туку задолжително содржат само атрибути.
- Ентитетите се Thread-Safe, а вредносните објекти најчесто не се.
- Еднаквоста меѓу два ентитета од иста класа се утврдува врз база на примарниот клуч.**
- Еднаквост меѓу два вредносни објекти од иста класа се утврдува врз база на еднаквоста на сите атрибути кои ги поседуваат**
- Поради нивната неменливост (immutability), вредносните објекти не се Thread-Safe
- Вредносните објекти треба да бидат неменливи (immutable) штом се направи нивна иницијализација.**
- Ентитетите најчесто содржат бизнис логика и сите методи се декларирали како static за да бидат thread-safe
- Вредносните објекти, освен атрибути, може да содржат бизнис логика.**

7. Сите врски кои имаат:

- **to-one** врска = FetchType.EAGER - ќе ги повлече сите негови врски заедно со него
- **to-many** врска = FetchType.LAZY
- **EntityGraph.EntityType.FETCH** – ако сите атрибути кои што ги специфицирам ќе бидат третирани со EAGER type, а сите атрибути кои што не ги специфицирам ќе бидат третирани со LAZY type
- **EntityGraph.EntityType.LOAD** - сите атрибути кои што ги специфицирам ќе бидат третирани со EAGER type, а сите атрибути кои што не ги специфицирам ќе бидат третирани како што е дефинирано во нивниот тип, односно она што е default доколку немаат специфицирано

Прашања од јуни практично 2021

/2021/L-38_33702 / Испит - ЈУНИ / Испит Јуни - практичен дел

Question 9
Answer saved
Marked out of 2.00
Flag question

Кои од следните искази се точни? (еден или повеќе точни одговори)

Select one or more:

Default fetch type за @OneToMany релација е FetchType.EAGER.
Кога во рамки на користењето на EntityGraph, type = EntityGraph.EntityGraphType.LOAD, атрибутите кои не се специфицирани ќе бидат третирани зависно нивниот специфициран или пак default FetchType.

Default fetch type за @OneToOne релација е FetchType.EAGER.
Default fetch type за @ManyToOne релација е FetchType.EAGER.
Кога во рамки на користењето на EntityGraph, type = EntityGraph.EntityGraphType.LOAD, атрибутите кои се специфицирани преку азимут на графот ќе бидат третирани како FetchType.EAGER, а останатите како FetchType.LAZY.

Give your reasons

A▼ B I ≡ ≡ % S

1.

-38_33702 / Испит - ЈУНИ / Испит Јуни - практичен дел

Question 10
Not yet answered
Marked out of 3.00
Flag question

Поврзете ги концептите!

Добавање на свойство @Version private Long version во рамки на одреден ентитет

Имплементирање на scheduled task

Публикување на одреден настан во рамки на сервисниот слој

Locks

Choose...
Choose...
Choose...
Choose...

Give your reasons

A▼ B I ≡ ≡ % S

2.

Optimistic locking uses version attributes included in entities to control concurrent modifications on them.

There are several rules which we should follow while declaring version attributes:

- a. each entity class must have only one version attribute
- b. it must be placed in the primary table for an entity mapped to several tables
- c. type of a version attribute must be one of the following: int, Integer, long, Long, short, Short, java.sql.Timestamp

<https://docs.spring.io/spring-framework/docs/3.1.x/spring-framework-reference/html/scheduling.html>

Публикувањо настани е некаква способност овозможена од Апликацискот контекст. Вушност, тоа е некој механизам кој што можеме да го искористиме за комуникација помеѓу компонентите во рамките на некоја Spring Boot апликација. Имаме PUBLISHER кој што праќа некакви пораки кои што претставуваат Events кои што некои listeners слушаат и чекаат такви Events. Тоа значи дека секоја една компонента може да биде Publisher и да публикува настан, додека друга компонента може да биде listener и да слуша дали има вакви настани и соодветно на тоа ако има ваков настан да реагира и да произведе некакво однесување во тој момент. Секој еден настан/Event генерално е јава објект кој што едноставно наследува од ApplicationEvent.

а трговија-2020/2021/L

702 / Испит - ЈУНИ / Испит Јуни - практични дел

Question 1
Not yet answered
Marked out of 4.00
Flag question

Give your reasons

Odberete gi soodvetnите i podredete ti arhitekturnite komponenti na eden ogranichen kontekst (bounded context):

External World <-> Input Adapter <-> Application Service <-> Domain Model <-> Output Adapter <-> Databases/Message Broker/Document Repositories

Events

Repositories

3.

```
graph LR; subgraph ExternalWorld [External World]; end; subgraph InputAdapter [Input Adapter]; end; subgraph ApplicationService [Application Service]; end; subgraph DomainModel [Domain Model]; end; subgraph OutputAdapter [Output Adapter]; end; subgraph DB [Databases/Message Broker/Document Repositories]; end; subgraph Events [Events]; end; subgraph Repositories [Repositories]; end; subgraph UI [User Interface Representations]; end; subgraph Security [Security]; end; subgraph Transactions [Transactions]; end; subgraph TaskCoordination [Task Coordination]; end; subgraph UseCaseController [Use Case Controller]; end; subgraph Entities [Entities]; end; subgraph BusinessLogic [Business Logic]; end; subgraph DomainEvents [Domain Events]; end; subgraph Repositories2 [Repositories]; end; subgraph Documents [Documents]; end; subgraph Cache [Cache]; end; subgraph Messaging [Messaging]; end; ExternalWorld <-> InputAdapter; InputAdapter <-> ApplicationService; ApplicationService <-> DomainModel; DomainModel <-> OutputAdapter; OutputAdapter <-> DB; UI -- "Security, Transactions, Task Coordination, Use Case Controller" --> InputAdapter; ApplicationService -- "Entities, Business Logic, Domain Events" --> DomainModel; DomainModel -- "Repositories, Documents, Cache, Messaging" --> OutputAdapter; InputAdapter -- "Domain Events" --> ApplicationService; ApplicationService -- "Persistence" --> DomainModel; DomainModel -- "Domain Events" --> OutputAdapter; OutputAdapter -- "Domain Events" --> DB;
```

Question 2
Not yet answered
Marked out of 4.00
Flag question

Нека е даден следниот код за **PetsController**. Потребно е истот да се дополни за да се обезбеди:

```

@RestController
@RequestMapping("/api")
public class PetsController {
    private final PetsService service;
    public PetsController(PetsService service) {
        this.service = service;
    }
    @GetMapping("/pets")
    public List<Pet> findAll() {
        return this.service.findAll();
    }
    @GetMapping("/pets/pagination")
    public List<Pet> findAllWithPagination(Pageable x) {
        return this.service.findAllWithPagination(x).getContent();
    }
    @GetMapping("/pet/{id}")
    public ResponseEntity<Pet> findById(@PathVariable long id) {
        return this.service.findById(id)
            .map(x -> ResponseEntity.ok().body(x))
            .orElseGet(() -> ResponseEntity.notFound().build());
    }
}

```

4.

ИМТ-2020/2021/L-38_33702 / Испит - ЈУНИ / Испит Јуни - практичен дел

Question 7
Not yet answered
Marked out of 3.00
Flag question

Што од понуденото е точно за доменските настани (Domain Events)?

Select one or more:

- a. Перзистирањето на промените во агрегатот и доменскиот настан најчесто не се прави во иста трансакција поради потенцијални проблеми со блокада.
- b. Содржат сопствен идентификационен број кој служи за нивно разликување од останатите настани.
- c. Домен настанит може по потреба да чуваат комплиексна бизнис логика.
- d. Не се Thread-Safe.
- e. Нивната состојба не е конзистентна и нивните атрибути може често да бидат променети по нивното креирање.
- f. Секогаш се неменливи (immutable) и Thread-Safe.
- g. Содржат временска марка за креирањето на настапот.

Give your reasons

5.

Точни: a,b, g,
se immutable

Question 8

Not yet
answered

Marked out of
5.00

Flag question

Изабери која од понудените опции дава валидна синтакса во ReactJS за прикажување на детали за предмет во една студиска програма. Речиси на компонентата е **Course**. Податоците за курсот се предаваат како својство на следниот начин:

```
let obj={  
    id:1, courseName:"ERT", courseFund:"2+2+1"  
};  
  
<Course course={obj}/>
```

Select one:

import React, {Component} from 'react'

class Course extends Component {
 render () {
 return (
 <li className="list-group-item">
 <div>
 <div className="row">
 <div className="col-md-2">
 {this.props.id}
 </div>
 <div className="col-md-5">
 {this.props.courseName}
 </div>
 <div className="col-md-4">
 {this.props.courseFund}
 </div>
 </div>
 </div>

);
 }
}

6.

Expression	Means
0 0 12 * * ?	Fire at 12:00 PM (noon) every day
0 15 10 ? * *	Fire at 10:15 AM every day
0 15 10 * * ?	Fire at 10:15 AM every day
0 15 10 * * ? *	Fire at 10:15 AM every day
0 15 10 * * ? 2005	Fire at 10:15 AM every day during the year 2005
0 * 14 * * ?	Fire every minute starting at 2:00 PM and ending at 2:59 PM, every day
0 0/5 14 * * ?	Fire every 5 minutes starting at 2:00 PM and ending at 2:55 PM, every day
0 0-5 14 * * ?	Fire every 5 minutes starting at 2:00 PM and ending at 2:55 PM, AND fire every 5 minutes starting at 6:00 PM and ending at 6:55 PM, every day
0 10,44 14 ? 3 WED	Fire every minute starting at 2:00 PM and ending at 2:05 PM, every day
0 15 10 ? * MON-FRI	Fire at 10:15 AM every Monday, Tuesday, Wednesday, Thursday and Friday
0 15 10 15 * ?	Fire at 10:15 AM on the 15th day of every month
0 15 10 L * ?	Fire at 10:15 AM on the last day of every month
0 15 10 ? * 6L	Fire at 10:15 AM on the last Friday of every month
0 15 10 ? * 6L 2002-2005	Fire at 10:15 AM on every last friday of every month during the years 2002, 2003, 2004, and 2005
0 15 10 ? * 6#3	Fire at 10:15 AM on the third Friday of every month
0 0 12 1/5 * ?	Fire at 12 PM (noon) every 5 days every month, starting on the first day of the month
0 11 11 11 11 ?	Fire every November 11 at 11:11 AM

Quiz navigation

Finish attempt ...

Question 1

Answer saved

Marked out of 1.00

Flag question

Кој од наведените cron изрази ќе ни помогне да дефинираме task кој ќе се извршува секој ден во 12 часот попладне (12 PM)?

Select one:

- a. 0 0 12 * * ?
- b. 0 12 0 * * ?
- c. 0 0 12 1 * ?
- d. 0 2 0 0 * * ?

0 0 12 * *

sec ; min ; hour ; day of month ; month ; day of week; year

[Clear my choice](#)

Quiz navigation

Finish attempt ...

Question 2

Answer saved

Marked out of 1.00

Flag question

Каква анотација недостасува на методот `takeNameAndPriceByProjection()` на сликата?

```
@Repository
public interface ProductRepository extends JpaRepository<Product, Long> {
    Optional<Product> findByName(String name);
    void deleteByName(String name);
    List<ProductProjection> takeNameAndPriceByProjection();
}
```

Select one:

- a. @Query
- b. не треба да има анотација
- c. @Query("select p.name, p.price from Product p")
- d. @Query("select * from Product")

[Clear my choice](#)

Quiz navigation

Finish attempt ...

Question 3

Answer saved

Marked out of 1.00

Flag question

Што недостига во рамки на кодот на сликата, со цел да класата `ProductCreatedEvent` претставува настан?

```
public class ProductCreatedEvent {
}
```

Select one:

- a. класата `ProductCreatedEvent` да се анотира со анотација `@Bean`
- b. класата `ProductCreatedEvent` да наследува од класата `ApplicationEvent`
- c. класата `ProductCreatedEvent` да се анотира со анотација `@Component`
- d. класата `ProductCreatedEvent` да се анотира со анотација `@Entity`

[Clear my choice](#)

Quiz navigation

Finish attempt ...

Question 4

Answer saved

Marked out of 1.00

Flag question

Доколку во рамки на еден сервис сакаме да публикуваме одреден настан, за тоа ни е потребно?

Select one:

- a. да инјектираме `ApplicationEventPublisher` и да го искористиме `publishEvent()` метод
- b. да инјектираме `ApplicationEvent`
- c. да инјектираме `ApplicationEventPublisher` и да го искористиме `publish()` метод

[Clear my choice](#)

ПОД а) ТОЧНО

Quiz navigation

Question 5
Answer saved
Marked out of 1.00
Flag question

Koj e **default fetch type** кога станува збор за **@ManyToOne** релација, а кога станува збор за **@ManyToMany** релација?

Select one:

- a. FetchType.LAZY, FetchType.EAGER
- b. FetchType.LAZY, FetchType.LAZY
- c. FetchType.EAGER, FetchType.EAGER
- d. FetchType.EAGER, FetchType.LAZY

[Clear my choice](#)

Quiz navigation

Finish attempt ...

Nека е даден кодот на скрипта. Кои промени во апликацијата ќе ни обезбедат точна функционалност, односно правилно обновување на материјализираниот поглед?

```
@Repository
public interface ProductsPerManufacturerViewRepository
    extends JpaRepository<ProductsPerManufacturerView, Long> {

    @Transactional
    @Modifying(clearAutomatically = true)
    @Query(value = "REFRESH MATERIALIZED VIEW public.products_per_manufacturers", nativeQuery = true)
    void refreshMaterializedView();
}
```

Select one or more:

- a. повикување на методот refreshMaterializedView() во рамки на web(presentation) слојот
- b. повикување на методот refreshMaterializedView() во рамки на одреден EventListener
- c. повикување на методот refreshMaterializedView() во рамки на сервисната логика
- d. повикување на методот refreshMaterializedView() во рамки на одреден scheduled task

Quiz navigation

Question 7
Not yet answered
Marked out of 1.00
Flag question

На скрипта е даден дел од кодот кој се наоѓа во **UserRepository** класата, која наследува од **JpaRepository**.
Која е разликата помеѓу двата методи **fetchAll()** и **loadAll()**, во однос на начинот на кој ќе се повлечат податоците од базата на податоци?

```
@EntityGraph(type = EntityGraph.EntityGraphType.FETCH,
            attributePaths = {"carts"})
@Query("select u from User u")
List<User> fetchAll();

@EntityGraph(type = EntityGraph.EntityGraphType.LOAD,
            attributePaths = {"carts", "discount"})
@Query("select u from User u")
List<User> loadAll();
```

Select one:

- a. Каде методот loadAll() атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.EAGER, а останатите како FetchType.LAZY.
Каде методот fetchAll() разликата е во тоа што оние атрибути кои не се специфицирани ќе бидат третирани зависно нивниот специфициран или пак default FetchType.
- b. Каде методот fetchAll() атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.LAZY, а останатите како FetchType.EAGER.
Каде методот loadAll() разликата е во тоа што оние атрибути кои не се специфицирани ќе бидат третирани зависно нивниот специфициран или пак default FetchType.
- c. Каде методот fetchAll() атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.LAZY, а останатите како FetchType.EAGER.
Каде методот loadAll() разликата е во тоа што оние атрибути кои не се специфицирани ќе бидат третирани како FetchType.LAZY, а останатите како FetchType.EAGER.
- d. Каде методот fetchAll() атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.EAGER, а останатите како FetchType.LAZY.
Каде методот loadAll() разликата е во тоа што оние атрибути кои не се специфицирани ќе бидат третирани зависно нивниот специфициран или пак default FetchType.

[Clear my choice](#)



Question 8
Answer saved
Marked out of
100
Flag question

Што треба да се додаде кон елингот **Product** на склада, со цел да се имплементира **Optimistic Locking**?

```
@Data  
@Entity  
public class Product {  
  
    @Id  
    @GeneratedValue(strategy = GenerationType.IDENTITY)  
    private Long id;  
  
    private String name;  
  
    private Double price;  
  
    private Integer quantity;  
  
    @ManyToOne  
    private Category category;  
  
    @ManyToOne  
    private Manufacturer manufacturer;  
  
    public Product() {  
    }  
}
```

Select one:

- a свойство:
private Long version;
- b. не може да се имплементира Optimistic Locking со промена на овој елинг
* c свойство:
@Version
private Long version;

[Clear my choice](#)

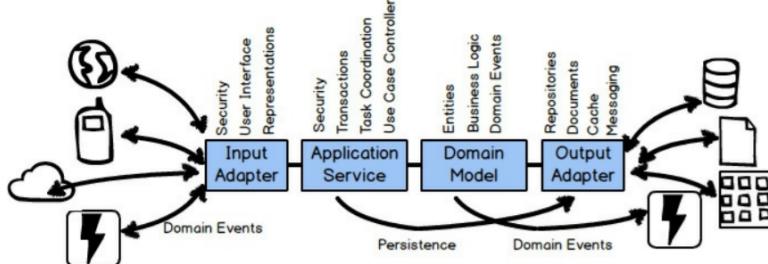
1. Опишете што се Drive by downloads преку пример и објаснете како може една организација да се заштити од нив?

2. Кои од понудение искази се точни во врска со Агрегати (Aggregates), Корен на агрегат (Aggregate Root)?
 - a. Промените кои треба да се рефлектираат во другите агрегати од страна на еден агрегат, треба да бидат постојано синхронизирани и во рамки на истата трансакција да се изврши нивната модификација.
 - b. **Трансакцискиот интегритет секогаш се одржува во рамки на агрегатот**
 - c. **Секој агрегат се состои од повеќе ентитети, каде што само еден ентитет се нарекува Корен на агрегатот**
 - d. Доменски настан (Domain Event) се праќаат само преку коренот на агрегатот.
 - e. Операциите кои предизвикуваат промена на состојбата се извршуваат во рамки на секој ентитет во агрегатот.
 - f. Секоја промена на локалните ентити во еден агрегат е независна и коренот на агрегатот нема потреба да е свесен за истата
 - g. **Состојбата на агрегатите задолжително мора да е постојано конзистентна и коренот на агрегатот е одговорен за спроведување на сите бизнис правила во рамки на агрегатот.**
 - h. **Еден агрегат може да е референциран од надворешниот свет само преку коренот на агрегатот.**

3. Одберете кои од понудените искази се точни во рамки на архитектурата на еден ограничен контекст.
 - Трансакциите се менаџираат во Domain Service
 - Безбедноста на надворешниот свет се менаџира во Output Adapter.
 - Безбедноста на надворешниот свет се менаџира во Input Adapter.
 - Трансакциите треба да ги менаџираат на ниво на Repository
 - Безбедноста во бизнис операциите се менаџира во Input Adapter
 - Трансакциите се менаџираат во Application Service**
 - Безбедноста во бизнис операциите се менажира во Domain Model
 -

Architecture

There is another question that you may have wondered about. What's inside a *Bounded Context*? Using this *Ports and Adapters* [IDDD] architecture diagram, you can see that a *Bounded Context* is composed of more than a domain model.



These layers are common in a *Bounded Context*: *Input Adapters*, such as user interface controllers, REST endpoints, and message listeners; *Application Services* that orchestrate use cases and manage transactions; the domain model that we've been focusing on; and *Output Adapters* such as persistence management and message senders. There is much to be said about the various layers in this architecture, and it is too elaborate to state in this distilled book. See [Chapter 4 of Implementing Domain-Driven Design](#) [IDDD] for an exhaustive discussion.

Technology-Free Domain Model

Although there will be technology scattered throughout your architecture, the domain model should be free of technology. For one thing, that's why transactions are managed by the application services and not by the domain model.

4. Што од понуденото е дел од ДДД дизајнот:

- Агрегати (Aggregates)** не сум сигурна
- Ограничени контексти (Bounded Contexts)** точно
- Доменски настани (Domain Events)** не сум сигурна
- Стратегија за маркетинг (Marketing Strategy)
- Сеопфатен јазик (Ubiquitous Language)** точно
- Искуство при користење (User Experience)
- Кориснички интерфејс (User Interface)
- Микросервисна архитектура

5. Наведете пример за вашата апликација и направете груба процена за буџетот потребен за е присуство, земајќи ги предвид најчестите ставки кои влагаат во буџетот на веб-сајтовите. При процената, земете во предвид дека и за софтверот и за хардверот ќе користите надворешно решение (outsourcing). При тоа, образложете кои се педностите на ваквите решенија за софтверот и хардверот во однос на спротивните опции (пр. outsourcing решение во однос на In-house решение)

6. Кои од следните изрази се точни за ентитети (Entities) и вредносни објекти (Value Objects) во ДДД?

- Вредносните објекти не смее да содржат бизнис логика, туку задолжително содржат само атрибути.
- Ентитетите се Thread-Safe, а вредносните објекти најчесто не се.
- Еднаквоста меѓу два ентитета од иста класа се утврдува врз база на примарниот клуч.**
- Еднаквост меѓу два вредносни објекти од иста класа се утврдува врз база на еднаквоста на сите атрибути кои ги поседуваат**
- Поради нивната неменливост (immutability), вредносните објекти не се Thread-Safe
- Вредносните објекти треба да бидат неменливи (immutable) штом се направи нивна иницијализација.**
- Ентитетите најчесто содржат бизнис логика и сите методи се декларирали како static за да бидат thread-safe
- Вредносните објекти, освен атрибути, може да содржат бизнис логика.**

7. Сите врски кои имаат:

- **to-one** врска = FetchType.EAGER - ќе ги повлече сите негови врски заедно со него
- **to-many** врска = FetchType.LAZY
- **EntityGraph.EntityType.FETCH** – ако сите атрибути кои што ги специфицирам ќе бидат третирани со EAGER type, а сите атрибути кои што не ги специфицирам ќе бидат третирани со LAZY type
- **EntityGraph.EntityType.LOAD** - сите атрибути кои што ги специфицирам ќе бидат третирани со EAGER type, а сите атрибути кои што не ги специфицирам ќе бидат третирани како што е дефинирано во нивниот тип, односно она што е default доколку немаат специфицирано

Прашања од јуни практично 2021

/2021/L-38_33702 / Испит - ЈУНИ / Испит Јуни - практичен дел

Question 9
Answer saved
Marked out of 2.00
Flag question

Кои од следните искази се точни? (еден или повеќе точни одговори)

Select one or more:

Default fetch type за @OneToMany релација е FetchType.EAGER.
Кога во рамки на користењето на EntityGraph, type = EntityGraph.EntityGraphType.LOAD, атрибутите кои не се специфицирани ќе бидат третирани зависно нивниот специфициран или пак default FetchType.

Default fetch type за @OneToOne релација е FetchType.EAGER.
Default fetch type за @ManyToOne релација е FetchType.EAGER.
Кога во рамки на користењето на EntityGraph, type = EntityGraph.EntityGraphType.LOAD, атрибутите кои се специфицирани преку азимут на графот ќе бидат третирани како FetchType.EAGER, а останатите како FetchType.LAZY.

Give your reasons

A▼ B I ≡ ≡ % S

1.

-38_33702 / Испит - ЈУНИ / Испит Јуни - практичен дел

Question 10
Not yet answered
Marked out of 3.00
Flag question

Поврзете ги концептите!

Добавање на свойство @Version private Long version во рамки на одреден ентитет

Имплементирање на scheduled task

Публикување на одреден настан во рамки на сервисниот слој

Locks

Choose...
Choose...
Choose...
Choose...

Give your reasons

A▼ B I ≡ ≡ % S

2.

Optimistic locking uses version attributes included in entities to control concurrent modifications on them.

There are several rules which we should follow while declaring version attributes:

- a. each entity class must have only one version attribute
- b. it must be placed in the primary table for an entity mapped to several tables
- c. type of a version attribute must be one of the following: int, Integer, long, Long, short, Short, java.sql.Timestamp

<https://docs.spring.io/spring-framework/docs/3.1.x/spring-framework-reference/html/scheduling.html>

Публикувањо настани е некаква способност овозможена од Апликацискот контекст. Вушност, тоа е некој механизам кој што можеме да го искористиме за комуникација помеѓу компонентите во рамките на некоја Spring Boot апликација. Имаме PUBLISHER кој што праќа некакви пораки кои што претставуваат Events кои што некои listeners слушаат и чекаат такви Events. Тоа значи дека секоја една компонента може да биде Publisher и да публикува настан, додека друга компонента може да биде listener и да слуша дали има вакви настани и соодветно на тоа ако има ваков настан да реагира и да произведе некакво однесување во тој момент. Секој еден настан/Event генерално е јава објект кој што едноставно наследува од ApplicationEvent.

а трговија-2020/2021/L

702 / Испит - ЈУНИ / Испит Јуни - практични дел

Question 1
Not yet answered
Marked out of 4.00
Flag question

Give your reasons

Odberete gi soodvetnите i podredete ti arhitekturnите komponenti na eden ogranichen kontekst (bounded context):

External World <-> Input Adapter <-> Application Service <-> Domain Model <-> Output Adapter <-> Databases/Message Broker/Document Repositories

Events

Repositories

3.

```
graph LR; subgraph ExternalWorld [External World]; end; subgraph InputAdapter [Input Adapter]; end; subgraph ApplicationService [Application Service]; end; subgraph DomainModel [Domain Model]; end; subgraph OutputAdapter [Output Adapter]; end; subgraph DB [Databases/Message Broker/Document Repositories]; end; subgraph Events [Events]; end; subgraph Repositories [Repositories]; end; subgraph UI [User Interface Representations]; end; subgraph Security [Security]; end; subgraph Transactions [Transactions]; end; subgraph TaskCoordination [Task Coordination]; end; subgraph UseCaseController [Use Case Controller]; end; subgraph Entities [Entities]; end; subgraph BusinessLogic [Business Logic]; end; subgraph DomainEvents [Domain Events]; end; subgraph Repositories2 [Repositories]; end; subgraph Documents [Documents]; end; subgraph Cache [Cache]; end; subgraph Messaging [Messaging]; end; ExternalWorld <-> InputAdapter; InputAdapter <-> ApplicationService; ApplicationService <-> DomainModel; DomainModel <-> OutputAdapter; OutputAdapter <-> DB; UI -- "Security, Transactions, Task Coordination, Use Case Controller" --> InputAdapter; ApplicationService -- "Entities, Business Logic, Domain Events" --> DomainModel; DomainModel -- "Repositories, Documents, Cache, Messaging" --> OutputAdapter; InputAdapter -- "Domain Events" --> ApplicationService; ApplicationService -- "Persistence" --> DomainModel; DomainModel -- "Domain Events" --> OutputAdapter; OutputAdapter -- "Domain Events" --> DB;
```

Question 2
Not yet answered
Marked out of 4.00
Flag question

Нека е даден следниот код за **PetsController**. Потребно е истот да се дополни за да се обезбеди:

```

@RestController
@RequestMapping("/api")
public class PetsController {
    private final PetsService service;
    public PetsController(PetsService service) {
        this.service = service;
    }
    @GetMapping("/pets")
    public List<Pet> findAll() {
        return this.service.findAll();
    }
    @GetMapping("/pets/pagination")
    public List<Pet> findAllWithPagination(Pageable x) {
        return this.service.findAllWithPagination(x).getContent();
    }
    @GetMapping("/pet/{id}")
    public ResponseEntity<Pet> findById(@PathVariable long id) {
        return this.service.findById(id)
            .map(x -> ResponseEntity.ok().body(x))
            .orElseGet(() -> ResponseEntity.notFound().build());
    }
}

```

4.

ИМТ-2020/2021/L-38_33702 / Испит - ЈУНИ / Испит Јуни - практичен дел

Question 7
Not yet answered
Marked out of 3.00
Flag question

Што од понуденото е точно за доменските настани (Domain Events)?

Select one or more:

- а. Перзистирањето на промените во агрегатот и доменскиот настан најчесто не се прави во иста трансакција поради потенцијални проблеми со блокада.
- б. Содржат сопствен идентификационен број кој служи за нивно разликување од останатите настани.
- в. Домен настанит може по потреба да чуваат комплиексна бизнис логика.
- г. Не се Thread-Safe.
- д. Нивната состојба не е конзистентна и нивните атрибути може често да бидат променети по нивното креирање.
- е. Секогаш се неменливи (immutable) и Thread-Safe.
- ж. Содржат временска марка за креирањето на настапот.

Give your reasons

5.

Question 8

Not yet
answered

Marked out of
5.00

Flag question

Изабери која од понудените опции дава валидна синтакса во ReactJS за прикажување на детали за предмет во една студиска програма. Речиси на компонентата е **Course**. Податоците за курсот се предаваат како својство на следниот начин:

```
let obj={  
    id:1, courseName:"ERT", courseFund:"2+2+1"  
};  
  
<Course course={obj}/>
```

Select one:

import React, {Component} from 'react'

class Course extends Component {
 render () {
 return (
 <li className="list-group-item">
 <div>
 <div className="row">
 <div className="col-md-2">
 {this.props.id}
 </div>
 <div className="col-md-5">
 {this.props.courseName}
 </div>
 <div className="col-md-4">
 {this.props.courseFund}
 </div>
 </div>
 </div>

);
 }
}

6.

Expression	Means
0 0 12 * * ?	Fire at 12:00 PM (noon) every day
0 15 10 ? * *	Fire at 10:15 AM every day
0 15 10 * * ?	Fire at 10:15 AM every day
0 15 10 * * ? *	Fire at 10:15 AM every day
0 15 10 * * ? 2005	Fire at 10:15 AM every day during the year 2005
0 * 14 * * ?	Fire every minute starting at 2:00 PM and ending at 2:59 PM, every day
0 0/5 14 * * ?	Fire every 5 minutes starting at 2:00 PM and ending at 2:55 PM, every day
0 0-5 14 * * ?	Fire every 5 minutes starting at 2:00 PM and ending at 2:55 PM, AND fire every 5 minutes starting at 6:00 PM and ending at 6:55 PM, every day
0 10,44 14 ? 3 WED	Fire every minute starting at 2:00 PM and ending at 2:05 PM, every day
0 15 10 ? * MON-FRI	Fire at 10:15 AM every Monday, Tuesday, Wednesday, Thursday and Friday
0 15 10 15 * ?	Fire at 10:15 AM on the 15th day of every month
0 15 10 L * ?	Fire at 10:15 AM on the last day of every month
0 15 10 ? * 6L	Fire at 10:15 AM on the last Friday of every month
0 15 10 ? * 6L 2002-2005	Fire at 10:15 AM on every last friday of every month during the years 2002, 2003, 2004, and 2005
0 15 10 ? * 6#3	Fire at 10:15 AM on the third Friday of every month
0 0 12 1/5 * ?	Fire at 12 PM (noon) every 5 days every month, starting on the first day of the month
0 11 11 11 11 ?	Fire every November 11 at 11:11 AM

Quiz navigation

Finish attempt ...

Question 1

Answer saved

Marked out of 1.00

Flag question

Кој од наведените cron изрази ќе ни помогне да дефинираме task кој ќе се извршува секој ден во 12 часот попладне (12 PM)?

Select one:

- a. 0 0 12 * * ?
- b. 0 12 0 * * ?
- c. 0 0 12 1 * ?
- d. 0 2 0 0 * * ?

0 0 12 * *

sec ; min ; hour ; day of month ; month ; day of week; year

[Clear my choice](#)

Quiz navigation

Finish attempt ...

Question 2

Answer saved

Marked out of 1.00

Flag question

Каква анотација недостасува на методот `takeNameAndPriceByProjection()` на сликата?

```
@Repository
public interface ProductRepository extends JpaRepository<Product, Long> {
    Optional<Product> findByName(String name);
    void deleteByName(String name);
    List<ProductProjection> takeNameAndPriceByProjection();
}
```

Select one:

- a. @Query
- b. не треба да има анотација
- c. @Query("select p.name, p.price from Product p")
- d. @Query("select * from Product")

[Clear my choice](#)

Quiz navigation

Finish attempt ...

Question 3

Answer saved

Marked out of 1.00

Flag question

Што недостига во рамки на кодот на сликата, со цел да класата `ProductCreatedEvent` претставува настан?

```
public class ProductCreatedEvent {
}
```

Select one:

- a. класата ProductCreatedEvent да се анотира со анотација @Bean
- b. класата ProductCreatedEvent да наследува од класата ApplicationEvent
- c. класата ProductCreatedEvent да се анотира со анотација @Component
- d. класата ProductCreatedEvent да се анотира со анотација @Entity

[Clear my choice](#)

Quiz navigation

Finish attempt ...

Question 4

Answer saved

Marked out of 1.00

Flag question

Доколку во рамки на еден сервис сакаме да публикуваме одреден настан, за тоа ни е потребно?

Select one:

- a. да инјектираме ApplicationEventPublisher и да го искористиме неговиот publishEvent() метод
- b. да инјектираме ApplicationEvent
- c. да инјектираме ApplicationEventPublisher и да го искористиме неговиот publish() метод

[Clear my choice](#)

ПОД а) ТОЧНО

Quiz navigation

Question 5
Answer saved
Marked out of 1.00
Flag question

Koj e **default fetch type** кога станува збор за **@ManyToOne** релација, а кога станува збор за **@ManyToMany** релација?

Select one:

- a. FetchType.LAZY, FetchType.EAGER
- b. FetchType.LAZY, FetchType.LAZY
- c. FetchType.EAGER, FetchType.EAGER
- d. FetchType.EAGER, FetchType.LAZY

[Clear my choice](#)

Quiz navigation

Finish attempt ...

Nека е даден кодот на скрипта. Кои промени во апликацијата ќе ни обезбедат точна функционалност, односно правилно обновување на материјализираниот поглед?

```
@Repository
public interface ProductsPerManufacturerViewRepository
    extends JpaRepository<ProductsPerManufacturerView, Long> {

    @Transactional
    @Modifying(clearAutomatically = true)
    @Query(value = "REFRESH MATERIALIZED VIEW public.products_per_manufacturers", nativeQuery = true)
    void refreshMaterializedView();
}
```

Select one or more:

- a. повикување на методот refreshMaterializedView() во рамки на web(presentation) слојот
- b. повикување на методот refreshMaterializedView() во рамки на одреден EventListener
- c. повикување на методот refreshMaterializedView() во рамки на сервисната логика
- d. повикување на методот refreshMaterializedView() во рамки на одреден scheduled task

Quiz navigation

Finish attempt ...

Question 7
Not yet answered
Marked out of 1.00
Flag question

На скрипта е даден дел од кодот кој се наоѓа во **UserRepository** класата, која наследува од **JpaRepository**.
Која е разликата помеѓу двата методи **fetchAll()** и **loadAll()**, во однос на начинот на кој ќе се повлечат податоците од базата на податоци?

```
@EntityGraph(type = EntityGraph.EntityGraphType.FETCH,
            attributePaths = {"carts"})
@Query("select u from User u")
List<User> fetchAll();

@EntityGraph(type = EntityGraph.EntityGraphType.LOAD,
            attributePaths = {"carts", "discount"})
@Query("select u from User u")
List<User> loadAll();
```

Select one:

- a. Каде методот loadAll() атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.EAGER, а останатите како FetchType.LAZY.
Каде методот fetchAll() разликата е во тоа што оние атрибути кои не се специфицирани ќе бидат третирани зависно нивниот специфициран или пак default FetchType.
- b. Каде методот fetchAll() атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.LAZY, а останатите како FetchType.EAGER.
Каде методот loadAll() разликата е во тоа што оние атрибути кои не се специфицирани ќе бидат третирани зависно нивниот специфициран или пак default FetchType.
- c. Каде методот fetchAll() атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.LAZY, а останатите како FetchType.EAGER.
Каде методот loadAll() разликата е во тоа што оние атрибути кои не се специфицирани ќе бидат третирани како FetchType.LAZY, а останатите како FetchType.EAGER.
- d. Каде методот fetchAll() атрибутите кои се специфицирани преку јазлите на графот ќе бидат третирани како FetchType.EAGER, а останатите како FetchType.LAZY.
Каде методот loadAll() разликата е во тоа што оние атрибути кои не се специфицирани ќе бидат третирани зависно нивниот специфициран или пак default FetchType.

[Clear my choice](#)



Question 8
Answer saved
Marked out of
100
Flag question

Што треба да се додаде кон елингот **Product** на склада, со цел да се имплементира **Optimistic Locking**?

```
@Data  
@Entity  
public class Product {  
  
    @Id  
    @GeneratedValue(strategy = GenerationType.IDENTITY)  
    private Long id;  
  
    private String name;  
  
    private Double price;  
  
    private Integer quantity;  
  
    @ManyToOne  
    private Category category;  
  
    @ManyToOne  
    private Manufacturer manufacturer;  
  
    public Product() {  
    }  
}
```

Select one:

- А свойство:
private Long version;
- Б не може да се имплементира Optimistic Locking со промена на овој елинг
● С свойство:
@Version
private Long version;

[Clear my choice](#)

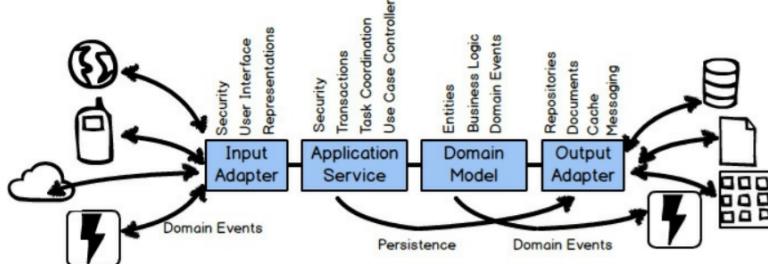
1. Опишете што се Drive by downloads преку пример и објаснете како може една организација да се заштити од нив?

2. Кои од понудение искази се точни во врска со Агрегати (Aggregates), Корен на агрегат (Aggregate Root)?
 - a. Промените кои треба да се рефлектираат во другите агрегати од страна на еден агрегат, треба да бидат постојано синхронизирани и во рамки на истата трансакција да се изврши нивната модификација.
 - b. **Трансакцискиот интегритет секогаш се одржува во рамки на агрегатот**
 - c. **Секој агрегат се состои од повеќе ентитети, каде што само еден ентитет се нарекува Корен на агрегатот**
 - d. Доменски настан (Domain Event) се праќаат само преку коренот на агрегатот.
 - e. Операциите кои предизвикуваат промена на состојбата се извршуваат во рамки на секој ентитет во агрегатот.
 - f. Секоја промена на локалните ентитети во еден агрегат е независна и коренот на агрегатот нема потреба да е свесен за истата
 - g. **Состојбата на агрегатите задолжително мора да е постојано конзистентна и коренот на агрегатот е одговорен за спроведување на сите бизнис правила во рамки на агрегатот.**
 - h. **Еден агрегат може да е референциран од надворешниот свет само преку коренот на агрегатот.**

3. Одберете кои од понудените искази се точни во рамки на архитектурата на еден ограничен контекст.
 - Трансакциите се менаџираат во Domain Service
 - Безбедноста на надворешниот свет се менаџира во Output Adapter.
 - Безбедноста на надворешниот свет се менаџира во Input Adapter.
 - Трансакциите треба да ги менаџираат на ниво на Repository
 - Безбедноста во бизнис операциите се менаџира во Input Adapter
 - Трансакциите се менаџираат во Application Service**
 - Безбедноста во бизнис операциите се менажира во Domain Model
 -

Architecture

There is another question that you may have wondered about. What's inside a *Bounded Context*? Using this *Ports and Adapters* [IDDD] architecture diagram, you can see that a *Bounded Context* is composed of more than a domain model.



These layers are common in a *Bounded Context*: *Input Adapters*, such as user interface controllers, REST endpoints, and message listeners; *Application Services* that orchestrate use cases and manage transactions; the domain model that we've been focusing on; and *Output Adapters* such as persistence management and message senders. There is much to be said about the various layers in this architecture, and it is too elaborate to state in this distilled book. See [Chapter 4 of Implementing Domain-Driven Design](#) [IDDD] for an exhaustive discussion.

Technology-Free Domain Model

Although there will be technology scattered throughout your architecture, the domain model should be free of technology. For one thing, that's why transactions are managed by the application services and not by the domain model.

4. Што од понуденото е дел од ДДД дизајнот:

- Агрегати (Aggregates)** не сум сигурна
- Ограничени контексти (Bounded Contexts)** точно
- Доменски настани (Domain Events)** не сум сигурна
- Стратегија за маркетинг (Marketing Strategy)
- Сеопфатен јазик (Ubiquitous Language)** точно
- Искуство при користење (User Experience)
- Кориснички интерфејс (User Interface)
- Микросервисна архитектура

5. Наведете пример за вашата апликација и направете груба процена за буџетот потребен за е присуство, земајќи ги предвид најчестите ставки кои влагаат во буџетот на веб-сајтовите. При процената, земете во предвид дека и за софтверот и за хардверот ќе користите надворешно решение (outsourcing). При тоа, образложете кои се педностите на ваквите решенија за софтверот и хардверот во однос на спротивните опции (пр. outsourcing решение во однос на In-house решение)

6. Кои од следните изрази се точни за ентитети (Entities) и вредносни објекти (Value Objects) во ДДД?

- Вредносните објекти не смее да содржат бизнис логика, туку задолжително содржат само атрибути.
- Ентитетите се Thread-Safe, а вредносните објекти најчесто не се.
- Еднаквоста меѓу два ентитета од иста класа се утврдува врз база на примарниот клуч.**
- Еднаквост меѓу два вредносни објекти од иста класа се утврдува врз база на еднаквоста на сите атрибути кои ги поседуваат**
- Поради нивната неменливост (immutability), вредносните објекти не се Thread-Safe
- Вредносните објекти треба да бидат неменливи (immutable) штом се направи нивна иницијализација.**
- Ентитетите најчесто содржат бизнис логика и сите методи се декларирали како static за да бидат thread-safe
- Вредносните објекти, освен атрибути, може да содржат бизнис логика.**

7. Сите врски кои имаат:

- **to-one** врска = FetchType.EAGER - ќе ги повлече сите негови врски заедно со него
- **to-many** врска = FetchType.LAZY
- **EntityGraph.EntityType.FETCH** – ако сите атрибути кои што ги специфицирам ќе бидат третирани со EAGER type, а сите атрибути кои што не ги специфицирам ќе бидат третирани со LAZY type
- **EntityGraph.EntityType.LOAD** - сите атрибути кои што ги специфицирам ќе бидат третирани со EAGER type, а сите атрибути кои што не ги специфицирам ќе бидат третирани како што е дефинирано во нивниот тип, односно она што е default доколку немаат специфицирано

Прашања од јуни практично 2021

/2021/L-38_33702 / Испит - ЈУНИ / Испит Јуни - практичен дел

Question 9
Answer saved
Marked out of 2.00
Flag question

Кои од следните искази се точни? (еден или повеќе точни одговори)

Select one or more:

Default fetch type за @OneToMany релација е FetchType.EAGER.
Кога во рамки на користењето на EntityGraph, type = EntityGraph.EntityGraphType.LOAD, атрибутите кои не се специфицирани ќе бидат третирани зависно нивниот специфициран или пак default FetchType.

Default fetch type за @OneToOne релација е FetchType.EAGER.
Default fetch type за @ManyToOne релација е FetchType.EAGER.
Кога во рамки на користењето на EntityGraph, type = EntityGraph.EntityGraphType.LOAD, атрибутите кои се специфицирани преку азимут на графот ќе бидат третирани како FetchType.EAGER, а останатите како FetchType.LAZY.

Give your reasons

A▼ B I ≡ ≡ % S

1.

-38_33702 / Испит - ЈУНИ / Испит Јуни - практичен дел

Question 10
Not yet answered
Marked out of 3.00
Flag question

Поврзете ги концептите!

Додавање на свойство @Version private Long version во рамки на одреден ентитет

Имплементирање на scheduled task

Публикување на одреден настан во рамки на сервисниот слој

Locks

Choose...
Choose...
Choose...
Choose...

Give your reasons

A▼ B I ≡ ≡ % S

2.

Optimistic locking uses version attributes included in entities to control concurrent modifications on them.

There are several rules which we should follow while declaring version attributes:

- a. each entity class must have only one version attribute
- b. it must be placed in the primary table for an entity mapped to several tables
- c. type of a version attribute must be one of the following: int, Integer, long, Long, short, Short, java.sql.Timestamp

<https://docs.spring.io/spring-framework/docs/3.1.x/spring-framework-reference/html/scheduling.html>

Публикувањо настани е некаква способност овозможена од Апликацискот контекст. Вушност, тоа е некој механизам кој што можеме да го искористиме за комуникација помеѓу компонентите во рамките на некоја Spring Boot апликација. Имаме PUBLISHER кој што праќа некакви пораки кои што претставуваат Events кои што некои listeners слушаат и чекаат такви Events. Тоа значи дека секоја една компонента може да биде Publisher и да публикува настан, додека друга компонента може да биде listener и да слуша дали има вакви настани и соодветно на тоа ако има ваков настан да реагира и да произведе некакво однесување во тој момент. Секој еден настан/Event генерално е јава објект кој што едноставно наследува од ApplicationEvent.

а трговија-2020/2021/L

702 / Испит - ЈУНИ / Испит Јуни - практични дел

Question 1
Not yet answered
Marked out of 4.00
Flag question

Give your reasons

Odberete gi soodvetnите i podredete ti arhitekturnite komponenti na eden ogranichen kontekst (bounded context):

External World <-> Input Adapter <-> Application Service <-> Domain Model <-> Output Adapter <-> Databases/Message Broker/Document Repositories

Events

Repositories

3.

```
graph LR; subgraph ExternalWorld [External World]; end; subgraph InputAdapter [Input Adapter]; end; subgraph ApplicationService [Application Service]; end; subgraph DomainModel [Domain Model]; end; subgraph OutputAdapter [Output Adapter]; end; subgraph DB [Databases/Message Broker/Document Repositories]; end; subgraph Events [Events]; end; subgraph Repositories [Repositories]; end; ExternalWorld <-> InputAdapter; InputAdapter <-> ApplicationService; ApplicationService <-> DomainModel; DomainModel <-> OutputAdapter; OutputAdapter <-> DB; InputAdapter -- "User Interface Representations" --> ApplicationService; InputAdapter -- "Security, Transactions, Task Coordination, Use Case Controller" --> ApplicationService; ApplicationService -- "Entities, Business Logic" --> DomainModel; ApplicationService -- "Domain Events" --> Events; ApplicationService -- "Repositories, Documents, Cache" --> OutputAdapter; ApplicationService -- "Messaging" --> DB; OutputAdapter -- "Persistence" --> DomainModel; OutputAdapter -- "Domain Events" --> DB; OutputAdapter -- "Messaging" --> DB;
```

Question 2
Not yet answered
Marked out of 4.00
Flag question

Нека е даден следниот код за **PetsController**. Потребно е истот да се дополни за да се обезбеди:

```
@RestController
@RequestMapping("/api")
public class PetsController {
    private final PetsService service;
    public PetsController(PetsService service) {
        this.service = service;
    }
    @GetMapping("/pets")
    public List<Pet> findAll() {
        return this.service.findAll();
    }
    @GetMapping("/pets/pagination")
    public List<Pet> findAllWithPagination(Pageable x) {
        return this.service.findAllWithPagination(x).getContent();
    }
    @GetMapping("/pet/{id}")
    public ResponseEntity<Pet> findById(@PathVariable long id) {
        return this.service.findById(id)
            .map(x -> ResponseEntity.ok().body(x))
            .orElseGet(() -> ResponseEntity.notFound().build());
    }
}
```

4.

ИМТ-2020/2021/L-38_33702 / Испит - ЈУНИ / Испит Јуни - практичен дел

Question 7
Not yet answered
Marked out of 3.00
Flag question

Што од понуденото е точно за доменските настани (Domain Events)?

Select one or more:

- а. Перзистирањето на промените во агрегатот и доменскиот настан најчесто не се прави во иста трансакција поради потенцијални проблеми со блокада.
- б. Содржат сопствен идентификационен број кој служи за нивно разликување од останатите настани.
- в. Домен настанит може по потреба да чуваат комплиексна бизнис логика.
- г. Не се Thread-Safe.
- д. Нивната состојба не е конзистентна и нивните атрибути може често да бидат променети по нивното креирање.
- е. Секогаш се неменливи (immutable) и Thread-Safe.
- ж. Содржат временска марка за креирањето на настапот.

Give your reasons

A B I \equiv % S

5.

Question 8

Not yet
answered

Marked out of
5.00

Flag question

Изабери која од понудените опции дава валидна синтакса во ReactJS за прикажување на детали за предмет во една студиска програма. Речиси на компонентата е **Course**. Податоците за курсот се предаваат како својство на следниот начин:

```
let obj={  
    id:1, courseName:"ERT", courseFund:"2+2+1"  
};  
  
<Course course={obj}/>
```

Select one:

import React, {Component} from 'react'

class Course extends Component {
 render () {
 return (
 <li className="list-group-item">
 <div>
 <div className="row">
 <div className="col-md-2">
 {this.props.id}
 </div>
 <div className="col-md-5">
 {this.props.courseName}
 </div>
 <div className="col-md-4">
 {this.props.courseFund}
 </div>
 </div>
 </div>

);
 }
}

6.

Question 1Not yet
answeredMarked out of
10.00

Flag question

Описете ти трендовите на веб дизајн прилагодлив за прикажување на мобилни уреди и наведете ги нивните предности. Кој тренд, според вас е најдобро решение за компанија која веќе е долго време присутна на пазарот и остварува големи приходи? Образложете.

Кои димензии за сигурност во електронската трговија се опфаќат со енкрипцијата? Образложете.

Од аспект на маркетинг, објаснете зошто е потребен систем за Customer Relationship Management

Question 2Not yet
answeredMarked out of
10.00

Flag question

Question 3Not yet
answeredMarked out of
10.00

Flag question

Што од понуденото побарува ДДД дизајнот:

Select one or more:

- a. Дефинирање на протокол на комуникација помеѓу ограничени контексти со користење на доменски настани (Domain Events)
- b. Дефиниција на основни CRUD операции кои треба да ги има секој ентитет.
- c. Детално дефинирање на мали кластери кои имаат специфична улога во рамки на ограничениот контекст наречени агрегати (Aggregates)
- d. Дефинирање на јазик кој ќе овозможува ефикасна тимска комуникација наречен Сеопфатен јазик (Ubiquitous Language)
- e. Подобрување на корисничко искуство при користење на софтверот.
- f. Дефинирање на строги правила за именување на ентитетите кои може единствено да ги разберат софтверските инженери.
- g. Поделба на доменските модели во помали кластери со користење на стратегиски шаблон наречен ограничен контексти (Bounded Contexts)
- h. Строга дефиниција на кориснички интерфејс кој ќе биде лесно разбиралив од корисниците.

Question 4Not yet
answeredMarked out of
10.00

Flag question

Кои од понудените искази се точни во врска со Агрегати (Aggregates), Корен на агрегат (Aggregate Root)?

Select one or more:

- a. Состојбата на агрегатите задолжително мора да е постојано конзистентна и коренот на агрегатот е одговорен за спроведување на сите бизнис правила во рамки на агрегатот.
- b. Операциите кои предизвикуваат промена на состојбата се извршуваат во рамки на секој ентитет во агрегатот.
- c. Секој агрегат се состои од повеќе ентитети, каде што само еден ентитет се нарекува Корен на агрегатот.
- d. Промените кои треба да се рефлектираат во другите агрегати од страна на еден агрегат, треба да бидат постојано синхронизирани и во рамки на истата трансакција да се изврши нивната модификација.
- e. Еден агрегат може да е референциран од надворешниот свет само преку Коренот на агрегатот.
- f. Трансакцискиот интегритет секогаш се одржува во рамки на агрегатот.
- g. Доменски настан (Domain Event) се практикат само преку коренот на агрегатот.
- h. Секоја промена на локалните ентитети во еден агрегат е независна и коренот на агрегатот нема потреба да е свесен за истата.

Question 6Not yet
answeredMarked out of
10.00[Flag question](#)

Асоцирајте ги соодветните компоненти со соодветниот слој од архитектурата на еден ограничен контекст (bounded context):

Влезен адаптер - ,

Апликациски сервис - ,

Доменски модел - ,

Излезен адаптер - ,

Трансакции	Безбедност на апликациските сервиси и авторизација
Ентитети	Безбедност на REST сервис
Размена на пораки со други ограничен контексти	Кеш
Серијализација на податочни структури во JSON/XML формат	Бизнес логика

Кои од следните изрази се точни за ентитети (Entities) и вредносни објекти (Value Objects) во ДДД?

Select one or more:

- a. Репозиториумот е контејнер на вредносни објекти.
- b. Еднаквост меѓу два вредносните објекти од иста класа се утврдува врз база на еднаквоста на сите атрибути кои ги поседуваат.
- c. Вредносните објекти, освен атрибути, може да содржат бизнис логика.
- d. Поради нивната неменливост (immutability), вредносните објекти не се Thread Safe.
- e. Промената на вредносните објекти секогаш задолжително се логира во логот на настани.
- f. Неменливоста е задолжителна карактеристика на вредносните објекти.
- g. Комплексната бизнис логика се чува во ентитетите, додека вредносните објекти содржат едноставни проверки на доделени вредности на атрибути.
- h. Два ентитета од иста класа се исти доколку имаат иста вредност на примарниот клуч.
- i. Промената на состојбата на вредностите објекти се прави преку соодветен репозиториум (Repository).
- j. Сите атрибути на ентитетот се менуваат со едноставни setter методи.

Question 7Not yet
answeredMarked out of
10.00[Flag question](#)

1. Што се Data Breaches и како може една компанија да се заштити од нив?

-Претставува кражба на информации од одредени системи.Овде најчесто се користат багови, невнимателности преку кои се крадат самите информации за корисниците.Најголема data breach е Apollo со околу 9 милијарди кражби на корисници. Се користат најразлични механизми за како ќе се украдат информациите.Дел преку багови, дел преку social engineering. Најчесто се крадат информации од компании и се губи доверба на самите организации и голем број на корисници бидејќи нивните податоци се украдени. Од Data Breaches можеме да се заштитиме на повеќе начини, еден од нив е енкрипција на податоците, кеширање на пасворди затоа што не е исплатливо кеширани пасворди да се хакираат. Доколку имаме ваков data breach потребно е да се известат сите корисници за овој напад, да ги сменат своите пасворди што посекоро. Исто така причини поради кои може да настане ваков напад се хакирање користење на одредени пропусти поради не доволно заштита, користење на одредени багови во софтверите. Неавторизиран пристап, кој на пример може да настане преку phishing од страна на вработените, или поради некој слаб пасворд. Чести напади се исто така од страна на вработените и нивна невнимателност, доколку по грешка се комитува некој приватен клуч за најава. Постои уште една причина за напад, односно внатрешна кражба. Некој од вработените во фирмата да изврши кражба за лично збогатување.

2.Поврзете ги различните начини на комуникација на online marketing

- Подтипови на online marketing:

1)Affiliate marketing

-Доколку некој клиент купи продукт од нашиот веб сайт, ние добиваме пари.

2)Viral marketing

-Кога дизајнираме некакви реклами или пораки кои што се интересни за корисниците и главната цел е да се споделуваат помеѓу корисниците.

3)Lead Generation marketing

-Да ги донесе корисниците директно од точка А во точка Б, со претходно скокање на неколку чекори. Се со цел да го натера корисникот да купи нешто.

- Начини на комуникација:

1) One-to-one

- Комуницираме директно само со еден корисник преку мејл или фејсбук порака

2) Behavioral targeting

- На корисникот му нудиме продукти за кои од претходно претпоставуваме дека е заинтересиран и би купил.

3) Retargeting

- На повеќе начини и обиди пробуваме да го убедиме корисникот да купи одреден продукт.

4) Customization и customer co-production

-Доколку клиентот не е задоволен од одреден продукт и за истиот не известува, ни кажува за каква промена станува збор, а од наша страна ја извршуваме истата промена со цел да се задоволи клиентот. На тој начин и клиентот има удел во co-production.

5) Customer service

- Нуѓи директен или автоматизиран online chat и пристап до најчесто поставувани прашања (како за shipping, цени и.т.н)

3. Наведете пример за ваша апликација и направете груба процена за буџет потребен за е-присуство, земајќи ги предвид најчестите ставки кои влагаат во буџетот на веб-сајтовите. При процената, земете во предвид дека и за софтверот и за хардверот ќе користите ваше решение (in house). При тоа, образложете кои се предностите на ваквите решенија за софтверот и хардверот во однос на спротивните опции(пр. Ваше решение во однос на надворешно решение).

-Одлучуваме да направиме сајт и првично што треба да направиме е да купиме домайн име кое ќе го плаќаме на годишно ниво. Мора да имаме сертификат дека ние сме таа компанија, а не некој да се претставува во наше име и тој се плаќа на годишно ниво во однос на колку поддомайн имиња имаме. Со **in-house** ние имаме целосна контрола врз нашиот сајт, ние го одржуваме, ние ги правиме процесите според нашите потреби, може било која функционалност да се имплементира. Но, бидејќи со **in-house** започнуваме целосно од почеток, имаме повеќе багови и ни одзема повеќе време. Во зависност со колкав тим работиме, колкава е големината на проектот и колкаво време ни е потребо, треба да ги плаќаме програмерите за сатница, од прилика 25000 евра за да се направат основните функционалности, да се одржува серверот и да се наплатат сатниците за програмерите. Во однос за хардверот, ние самите треба да купиме сервер со добри перформанси кој ќе треба самите да го конфигурираме. Потребен ќе ни е линк до провајдер. Треба сами да се грижиме за проблемите при пад на струја. **Outsourcing** ни е побрзо и побезбедно. Тука не мора да се грижиме за хардверот премногу, бидејќи од нивна страна имаме веќе и сервер, домайн, сертификат и база и немаме потреба да инсталураме нешто. На пример можеме да ги користиме серверите на Cloud computing, а доколку сакаме да имаме целосна контрола тоа може да го правиме со користење на VPS (дополнителен рам, хард диск и повеќе јадрен процесор).

4. SWOT анализа на фирмa како Амазон за наш регион маркетинг и рекламирање за start up со едукативни содржини

- Предности за нашата фирмa се тоа што нуди уникатни едукативни содржини од поширок спектар кои се достапни единствено на нашиот сајт на државно ниво. Нашиот веб сајт е достапен 24/7.

Лесно и едноставно може да се изврши купувањето, споредбата и предложувањето на продукти. Нуѓи лесна навигација и подобра персонализација.

-Слабите страни на нашата фирма би биле тоа што се чека 10 дена за достава и цената за достава варира спрема големината на продуктот. Бидејќи многу од луѓето се скептични да плаќаат online, исплатата ја извршуваме во готово. Нашиот веб сайт има премногу обичен дизајн.

-Можностите на нашата фирма се достапност да не контактираат преку социјални мрежи, live chat со клиентите на веб сајтот, се следат чекорите на клиентите за да може да им се предложат слични продукти од нивниот интерес.

- Заканите на нашата фирма се нашите конкурентски фирмии кои што вршат достава на глобално ниво и нудат повеќе продукти од нас. Исто така нашите продукти може некој да ги препродава на eBay, Amazon и слично за странски пазари.

5. Како ги заштитуваме корисниците од credit card theft?

- Дигитален потпис
- Мулти фактор автентикација
- Fingerprint идентификација
- Two factor автентикација

6. Кое прашање одговара на интегритетот?

- Has information I transmitted or received been altered? - Клиент
- Has data on the site been altered without authorization? Is data being received from customers valid? Продавач

7. Кои процеси ги користиме за заштита на кредитни картички?

- Пред внесување на нашите податоци од кредитната картичка, треба внимателно да го провериме URL-то на payment провайдерот се со цел да се осигураме дека не е некаква форма на phishing. Не треба да внесуваме наши податоци на сомнителни страни или страни кои што немаат многу feedback, односно тоа треба да го правиме на страни на кои имаме доверба. Банките внесуваат повеќе начини на заштита како што се two factor автентикација, односно ни испраќаат код за верификација што треба да го внесиме пред да ја извршиме наплатата. Исто така имаме и други начини како дигитален потпис, мулти фактор автентикација и fingerprint идентификација пред извршување на наплатата. Сепак што повеќе заштита, толку е покомплицирано да се изврши таа наплата.

8. Што се вируси и како компанијата може да се заштити од нив?

- Вирусите се компјутерски програми кои што можат да се реплицираат или да направат копии од себеси, да се прикачат на друг програм или да се прошират во повеќе фајлови. Исто така можат да бидат прикачени на фајл за кој што корисникот ќе треба да изврши одредена акција за истиот да се активира. За корисникот да може да се заштити од вируси, треба често да прави ажурирање на оперативниот систем, firewall, да симнува одредени updates или апликации само од официјални страни, а не од илегални торенти кои што можат да содржат вируси.

9. 5 чекори при купување на лаптоп

- Awareness of need
- Search for more information
- Evaluation of alternatives
- Actual purchase decision
- Post-purchase contact with firm

10. Која метрика го мери времето на задржување на корисниците на сајтот?

- Conversion to customer metrics

11. Што претставува price discrimination?

- Во зависност од тоа кому што продаваме, цената се менува. На пример авио карти или резервации во хотел.

12. Каков систем е PayPal?

- Универзален стандард, мобилна платформа за плаќање.

13. Man in the middle со пример да се објасни и како да се заштитат компаниите

- Man in the middle е напад со пресретнување, напад во јавна мрежа. Настанува со ARP Poisoning, односно со лажни arp request и response и default gateway не е повеќе вистинскиот рутер, туку самиот напаѓач и со лажни сертификати се претставува во наше име и ги прислушкува нашите разговори, шпиунира, може да замени сертификати и да злоупотреби наши пораки. Во вакви ситуации треба внимателно да гледаме дали сертификатот или потписот се совпаѓа со она што е добиено на наша страна.

14. Персонализација

- Personalization припаѓа во one-to-one маркетинг. Означува можност да ги третираме луѓето врз основа на нивните искуства со нашиот веб сајт. Дава можност за промена на продуктот според барањата на клиентот. Со колачиња ги добиваме информациите потребни за нашите клиенти.

15. Приватност

- Со Privacy Policy имаме правила кои што корисниците треба да ги следат и да ги почитуваат се со цел за да си ги заштитат своите лични информации, а за лицата со попреченост имаме посебни правила кои им даваат до знаење дека можат безбедно да влезат на сајтот.

Can I control the use of information about myself transmitted to an e-commerce merchant? Клиент

What use, if any, can be made of personal data collected as part of an ecommerce transaction? Is the personal information of customers being used in an unauthorized manner? Продавач

16. DOS напад

- DOS напад означува преплавување на страната од страна на еден компјутер, со непотребни pings и page requests, односно испраќање на повеќе барања до серверот отколку што може да прими. Ова може да предизвика исклучување на сите веб сервери на сајтот. Овој напад е придружен и со blackmail обиди. Исто така има и botnets.

17. Кои се intervening variables во генералниот модел на однесување на купувачите?

- Market Stimuli, Social Networks, Communities.

18. Кои од наведените не се платформи за креирање веб-страни:

1) WordPress

2) Apache

3) Не ми текнува

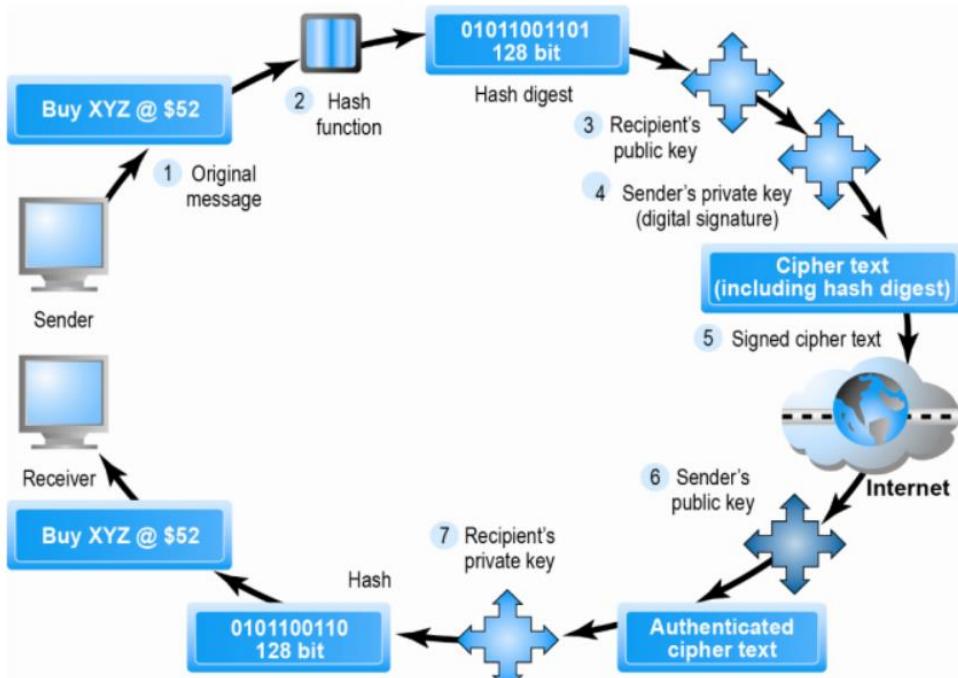
4) Google Sites

19. Што е Trojan, пример, како да се заштити компанија од него?

- Тројан е начин на влез на вирусот кој што може подоцна да ги инфицира сите фајлови на нашиот систем. Кога симнуваме некаква апликација или програма од илегална страна, од торент, можеме да симнеме заедно со тоа и тројански коњ кој може да го инфицира нашиот систем.

20.

Public Key Cryptography with Digital Signatures



Чекори:

- Испраќачот применува математички алгоритам (хеш функција) на порака и потоа ја енкриптира пораката и хеш резултатот (ги заклучува) со јавниот клуч на примателот.
- Испраќачот потоа ја енкриптира пораката и хеш резултатот со приватниот клуч на примателот - креирајќи дигитален потпис - за автентичност и неотфрање.
- Примачот првин го користи јавниот клуч на испраќачот за да ја автентицира пораката и потоа примачот го користи својот приватен клуч за да ги декриптира хеш резултатот и пораката.

21. Дали моделирањето на бизнис операциите е поддржано од ДДД дизајнот?

-ДДД и тактичките алатки не поддржуваат моделирање на бизнис операции, за разлика од стратегиските кои се стриктно наменети за тоа.

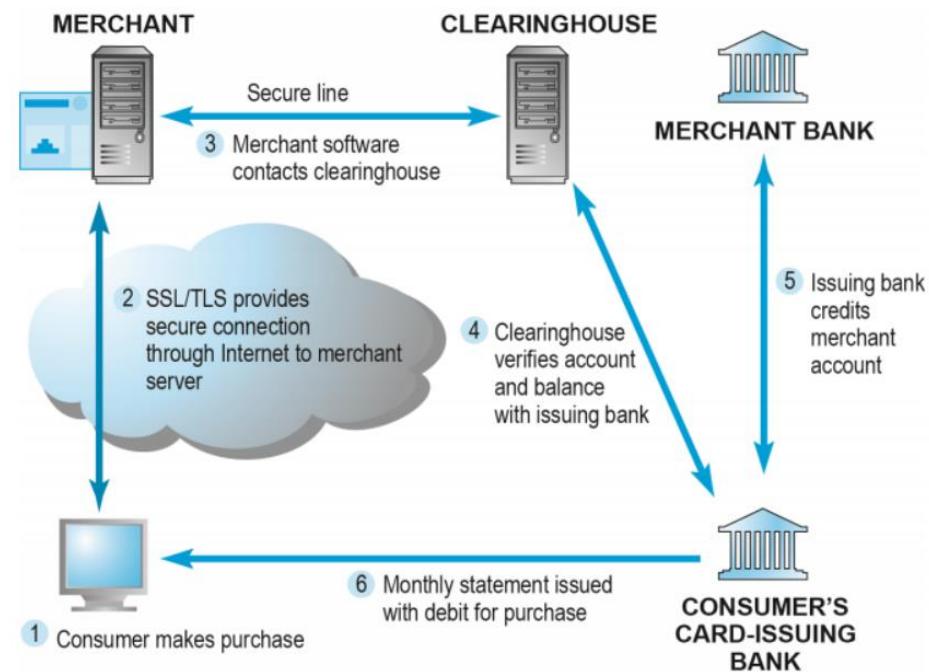
22. Наведи ги и објасни ги тензиите меѓу сигурноста и другите вредности (пр 5, слайд 8 во презентација 5-7)

-Ниво на употреба. Колку што има повеќе сигурносни мери, толку потежок станува сајтот за користење и толку поспор.

-Јавна безбедност и криминал на интернет. Користењето на технологијата од страна на криминалците за да се планираат криминални или заканување на безбедноста на нацијата/државата.

23.

How an Online Credit Transaction Works



Developing an E-commerce Security Plan

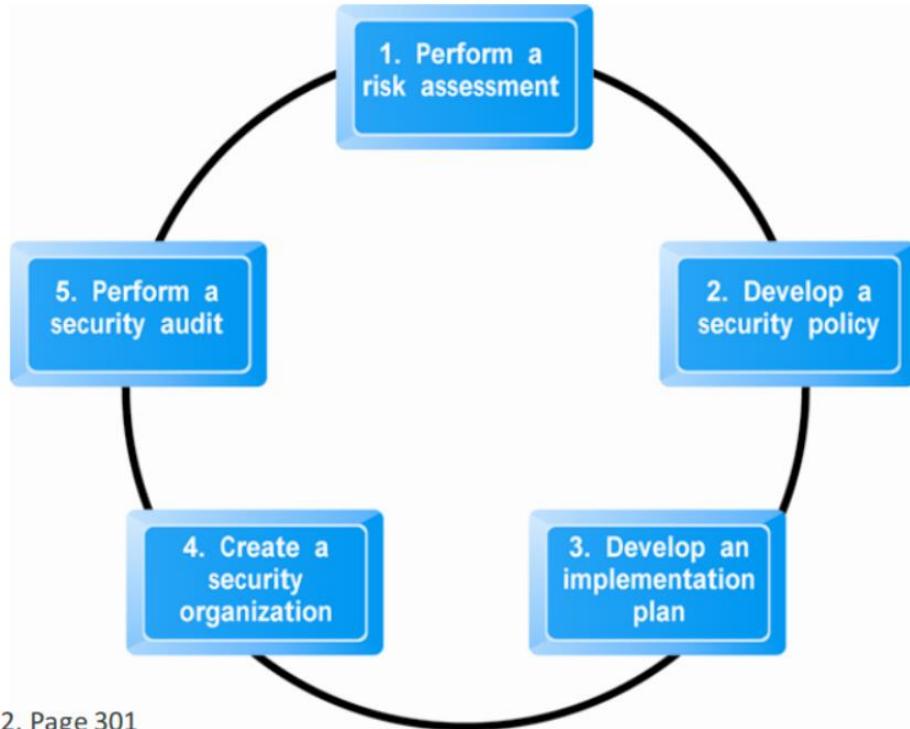


Figure 5.12. Page 301

Наведи ги и објасни ги чекорите при развој на e-commerce security plan

чекор 1: Согледај ги и научи сите можни ризици (Risk Assessment)

-Тоа е процес да се идентификуваат потенцијалните опасности и да се анализира што би можело да се случи доколку некоја од тие потенцијални опасности би се случиле.

Чекор 2: Развиј полиса на сигурност

-Полиса на сигурност е дефиниција за тоа што значи еден систем да е безбеден.

Чекор 3: Развиј имплементациски план

-Имплементацискиот план вклучува Безбедносна организација, контроли на пристап, процедури за автентикација како и биометрика, авторизациски полиси и авторизациски менаџмент системи.

Чекор 4: Организирај безбедносна околина

Чекор 5: Ревидирај ги и инспектирај безбедносните функционалности на сајтот

-На крај, добро би било да проба системот да се стави против првите можни претпоставени ризици за да се увидат добиените резултати.

25. Кои од понудените концепти припаѓаат на ДДД дизајнот?

- ❑ Data Access Object (DAO)
- ❑ Data Transfer Object (DTO)
- ❑ **слоевита архитектура**
- ❑ Model View – View Model
- ❑ **агрегати**
- ❑ **Ограничени контексти (Bounded Contexts)**
- ❑ model view controller (MVC)

26. Што од понуденото е точно за доменските настани (Domain Events)?

- ❑ Доменските настани може да се публикуваат преку презентацијскиот слој на апликацијата.
- ❑ Не содржат идентификатор, тој е карактеристичен само за ентитетите.
- ❑ Домен настаните може по потреба да чуваат комплексна бизнис логика.
- ❑ **Аплицирањето на промените во агрегатот и зачувувањето на доменскиот настан секогаш се прават во рамки на една трансакција.**
- ❑ **Содржат временска марка за креирањето на настанот.**
- ❑ Нивната состојба не е конзистентна и нивните атрибути може често да бидат променети по нивното креирање.
- ❑ **Доменските настани се корисни доколку постои механизам на нивна надежна достава до оние кои се препратиле на истите.**

27. Кои од следните изрази се точни за ентитети (Entities) и вредносни објекти (Value Objects) во ДДД?

- ☒ Вредносните објекти живеат самостојно и не зависат од останатите ентитети.
- ☒ Еднаквост меѓу два вредностни објекти од иста класа се утврдува врз база на еднаквоста на сите атрибути кои ги поседуваат.
- ☒ За ентитетите се чува историска информација за промени.
- ☒ Сите атрибути во ентитетите не треба да се менливи (Immutable) по нивната иницијализација.
- ☒ Вредносните објекти не смее да содржат бизнис логика, туку задолжително содржат само атрибути.
- ☒ Еднаквоста меѓу два ентитета од иста класа се утврдува врз база на примарен клуч.
- ☒ Вредносните објекти не живеат самостојно, туку најчесто припаѓаат на некој ентитет.
- ☒ Во вредносните објекти се чува комплексна бизнис логика и се оркестрира менацирањето на ентитетите.

28. Што од следново е точно за JWT токените?

- ☒ JWT е кратенка од JSON Web Token
- ☒ Сетирањето на безсостојбена сесија во Spring Security се прави со следната конфигурација:
`sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS)`
- ☒ Најчесто се корисни при авторизација на корисници, но и за безбедна размена на информации.
- ☒ Содржината на заглавјето кога се пристапува заштитен ресурс треба да го користи следниот формат: **Authorization: Bearer <token>**
- ☒ Се состојат од три дела: заглавје (header), приватен клуч (private key) и потпис (signature)
- ☒ JWT токените се најчест избор кога сакаме да користиме безсостојбена

автентикација.

☒ Најчесто се корисни при авторизација на корисници, но се лош избор при потреба од безбедна размена на информации.

☒ **Се состојат од три дела: заглавје (header), податоци (payload) и потпис (signature)**

29. Изберете што е точно од понуденото земајќи го предвид следното сценарио:

Потребно е да изработите систем кој прави пресметки на цената на автомобил врз база на цена на деловите од кои е составен. Дополнително, се води сметка и за бројот на автомобилски делови кои стојат на располагање во магацин. Ентитетите кои се користат во сценариото се следните: Car, CarPart и Warehouse.

☒ Бизнис логиката за пресметката на цената на автомобилот треба да биде поствена во CarPart ентитетот.

☒ Ентитетот CarPart е корен на сопствениот агрегат

☒ **Бизнис логиката за пресметка на цената на автомобилот треба да биде поствена во Car ентитетот.**

☒ **Во сценариото постојат два агрегати, еден ги содржи ентитетите Car и CarPart, другиот го содржи само ентитетот Warehouse.**

☒ **Ентитетот Car е корен на сопствениот агрегат.**

☒ Постојат три агрегати, посебен за секој ентитет. Секој од ентитетите е сопствен корен на агрегатот.

30. Кои од понудените искази се точни во врска со Агрегати (Aggregates), Корен на агрегат (Aggregate Root), Доменски настани (Domain Events)?

☒ **Промените кои треба да се рефлектираат во другите агрегати од страна на еден агрегат се прави преку концептот на Eventual consistency.**

☒ **Состојбата на агрегатите задолжително мора да е постојано конзистентна и коренот на агрегатите е одговорен за спроведување на сите бизнис правила во рамки на агрегатот.**

- ❑ Еден агрегат може да има повеќе корени (roots).
 - ❑ Трансакцискиот интегритет секогаш се одржува во рамки на цел ограничен контекст (Bounded Context).
 - ❑ **Доменските настани може да се пратат преку коренот на агрегатот и доменските сервиси.**
 - ❑ Промените кои треба да се рефлектираат во другите агрегати, од страна на еден агрегат, треба да бидат постојано синхронизирани и во рамки на истата трансакција да се извржи нивната модификација
 - ❑ **Еден агрегат може да е референциран од надворешниот свет само преку коренот на агрегатот.**
 - ❑ Секоја промена на локалните ентитети во еден агрегат е независна и коренот на агрегатот нема потреба да е свесен за истата.
- Трансакцискиот интегритет секогаш се одржува во рамки на агрегатот**

31. Што од понуденото е дел од ДДД дизајнот:

- ❑ Кориснички интерфејс (User Interface)
- ❑ Искуство при користење (User Experience)
- ❑ **Сеопфатен јазик (Ubiquitous Language)**
- ❑ **Ограничени контексти (Bounded Contexts)**
- ❑ **Доменски настани (Domain Events)**
- ❑ Микросервисна архитектура
- ❑ Стратегија за маркетинг (Marketing Strategy)
- ❑ **Агрегати (Aggregates)**

32. Кои од следните изрази се точни за ентитети (Entities) и вредносни објекти (Value Objects) во ДДД?

- ☒ Сите атрибути во ентитетите не треба да се менливи (Immutable) по нивната иницијализација.
- ☒ **Сите атрибути во вредносните објекти треба да бидат неменливи после иницијализација.**
- ☒ Во вредносните објекти се чува комплексната бизнис логика и се оркестрира менаџирањето на ентитетите.
- ☒ **Вредносните објекти немаат примарен клуч.**
- ☒ Ентитетите и вредносните објекти ја чуваат својата историја на менување во база.
- ☒ **Вредносните објекти немаат историја.**
- ☒ Ентитетите немаат историја.
- ☒ **Ентитетите имаат примарен клуч.**

Вредносните објекти, освен атрибути, може да содржат бизнис логика.

33. Кога најчесто се креира Anticorruption слој?

- ☒ **Кога е потребно да се овозможи изолација помеѓу два зависни домени кои имаат сопствен сеопфатен јазик (Ubiquitous Language).**
- ☒ **Кога има потреба од слој кој прави превод помеѓу два различни домена кои користат сопствени сеопфатни јазици (Ubiquitous Languages).**
- ☒ Кога постои јасно дефиниран протокол кој треба да се имплементира за да се направи интеграција помеѓу два домена.
- ☒ Кога преводот меѓу два сеопфатни јазици (Ubiquitous Languages) е невозможен и има потреба од директна интеграција на моделот на едниот домен во другиот.

34. Описете што се Drive by downloads преку пример и објаснете како може една организација да се заштити од нив?

Кога компјутерот ќе се зарази со злонамерен софтвер едноставно со посета на веб-страница, тој е познат како „drive by download“.

Овој тип на напад го добил името „ drive by download “, бидејќи корисникот не мора да застанува или да кликнува било каде на

злонамерната страница. Едноставно гледање на страницата е доволно за да предизвика инфекција што се случува во позадина и

без знаење или согласност на корисникот.

Во drive by download напад, криминалците компромитираат веб-страница, честопати легитимна, со вградување или инјектирање малициозни објекти во внатрешноста на веб-страниците. Инфекциите се

невидливи за корисникот и се движат од малициозен код на JavaScript до iFrames, врски, пренасочувања, малверзации,

скриптирање меѓу страници и други малициозни елементи.

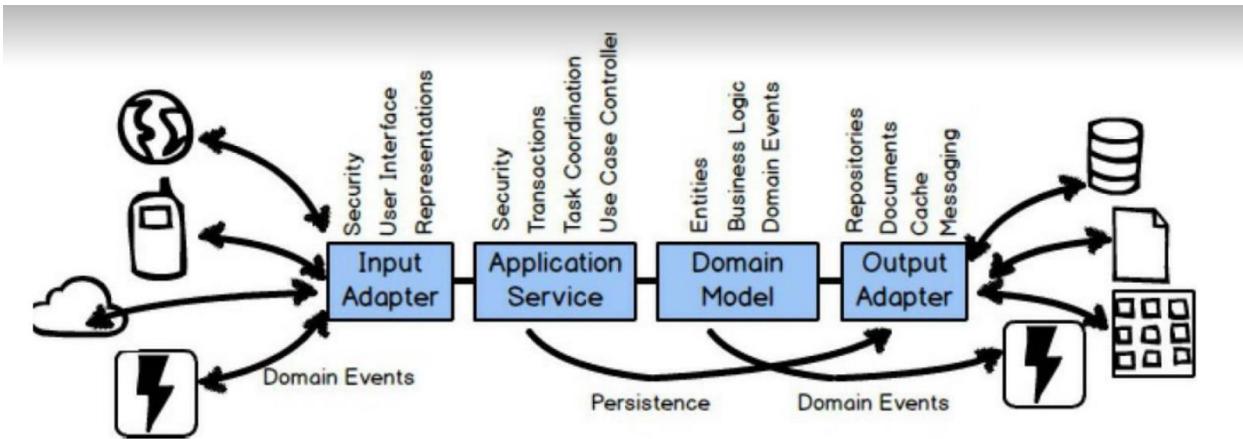
Кога корисникот посетува заразена веб -страница, прелистувачот на корисникот автоматски го чита малициозниот код, кој

веднаш го скенира компјутерот на жртвата за безбедносни слабости во оперативниот систем и други апликации.

Како да се заштитиме од таков тип на вируси:

- Update your software quickly and constantly
- Remove unnecessary software and plug-ins
- Stop using a privileged account for day-to-day work.
- Use a firewall
- Disable Java and JavaScript
- Use web-filtering software
 - Install an ad blocker

35. Одберете кои од понудените искази се точни во рамки на архитектурата на еден ограничен контекст.

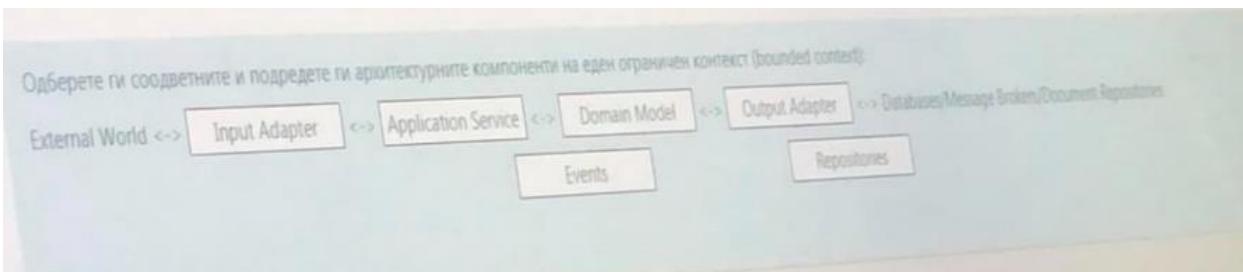


What's inside a Bounded Context ?

36. Што претставува публикување?

Публикувањето настани е некаква способност овозможена од Апликацискот контекст. Вушност, тоа е некој механизам кој што можеме да го искористиме за комуникација помеѓу компонентите во рамките на некоја Spring Boot апликација. Имаме PUBLISHER кој што праќа некакви пораки кои што претставуваат Events кои што некои listeners слушаат и чекаат такви Events. Тоа значи дека секоја една компонента може да биде Publisher и да публикува настан, додека друга компонента може да биде listener и да слуша дали има вакви настани и соодветно на тоа ако има ваков настан да реагира и да произведе некакво однесување во тој момент. Секој еден настан/Event генерално е јава објект кој што едноставно наследува од ApplicationEvent.

37. Одберете ги соодветните и подредете ги архитектурните компоненти на еден ограничен контекст



38. Што од понуденото е точно за доменските настани?

Што од понуденото е точно за доменските настани (Domain Events)?

Select one or more:

- a. Перзистирањето на промените во агрегатот и доменскиот настан најчесто не се прави во иста трансакција поради потенцијални проблеми со блокирање.
- b. Содржат сопствен идентификационен број кој служи за идентификација и разликување од останатите настани.
- c. Домен настани може по потреба да чуваат комплексна бизнис логика.
- d. Не се Thread-Safe.
- e. Нивната состојба не е конзистентна и нивните атрибути може често да бидат променети по нивното креирање.
- f. Секогаш се неменливи (Immutable) и Thread-Safe.
- g. Содржат временска марка за креирањето на настани.

Give your reasons

39. Доколку во рамки на еден сервис сакаме да публикуваме одреден настан, за тоа ни е потребно?

Доколку во рамки на еден сервис сакаме да публикуваме одреден настан, за тоа ни е потребно?

Select one:

- a. да инјектираме ApplicationEventPublisher и да го искористиме неговиот publishEvent() метод
- b. да инјектираме ApplicationEvent
- c. да инјектираме ApplicationEventPublisher и да го искористиме неговиот publish() метод

[Clear my choice](#)

под а) точно

40. Нека е даден кодот на сликата. Кои промени во апликацијата ќе ни обезбедат точна функционалност ,односно правилно обновување на материјализираниот поглед?

Нека е даден кодот на сликата. Кои промени во апликацијата ќе ни обезбедат точна функционалност, односно правилно обновување на материјализираниот поглед?

```
@Repository
public interface ProductsPerManufacturerViewRepository
    extends JpaRepository<ProductsPerManufacturerView, Long> {

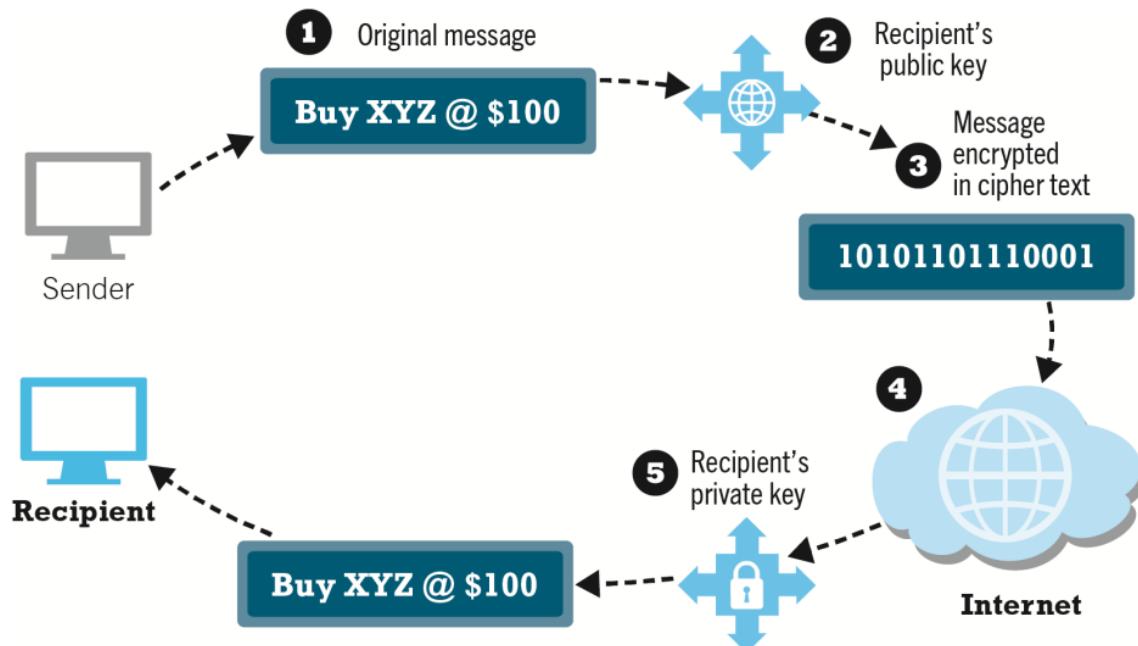
    @Transactional
    @Modifying(clearAutomatically = true)
    @Query(value = "REFRESH MATERIALIZED VIEW public.products_per_manufacturers", nativeQuery = true)
    void refreshMaterializedView();
}
```

Select one or more:

- a. повикување на методот refreshMaterializedView() во рамки на web(presentation) слојот
- b. повикување на методот refreshMaterializedView() во рамки на одреден EventListener
- c. повикување на методот refreshMaterializedView() во рамки на сервисната логика
- d. повикување на методот refreshMaterializedView() во рамки на одреден scheduled task

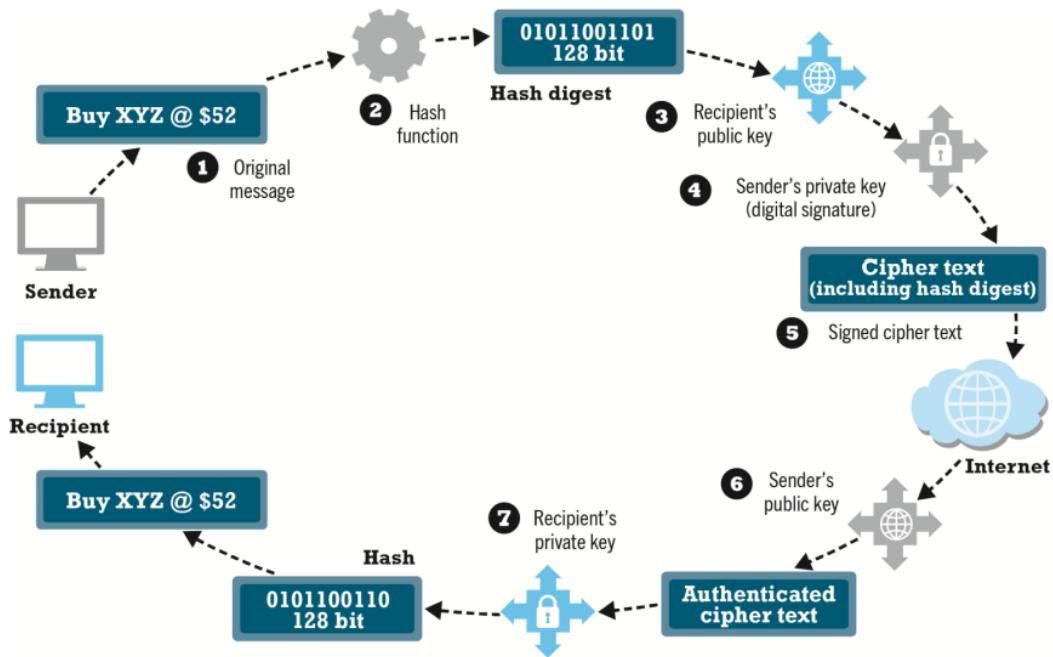
41. Обична криптографија

Figure 5.6 Public Key Cryptography: A Simple Case



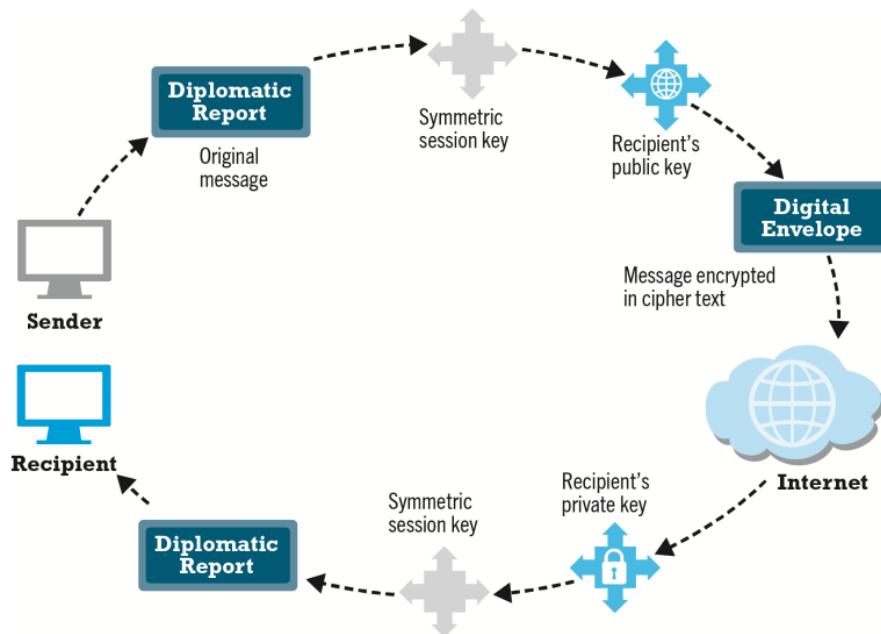
Copyright © 2020 Kenneth C. Laudon and Carol Guercio Traver

Figure 5.7 Public Key Cryptography with Digital Signatures



43. Криптографија со симетричен клуч – digital envelope дигитално плико

Figure 5.8 Public Key Cryptography: Creating a Digital Envelope



Copyright © 2020 Kenneth C. Laudon and Carol Guercio Traver