|  | УНИВЕРЗИТЕТ У НОВОМ САДУ  **ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У НОВОМ САДУ** |  |
| --- | --- | --- |

Исидора Савић

**Систем за интродукцију и реинтродукцију текуница**

Дипломски рад

- Основне академске студије -

Нови Сад, 2022.

Садржај

[1. Увод 4](#_Toc113547111)

[2. Стање у области 6](#_Toc113547112)

[3. Коришћене технике и технологијe 7](#_Toc113547113)

[3.1. *React* 7](#_Toc113547114)

[3.2. *REST* 8](#_Toc113547115)

[3.3. *Spring Boot* 8](#_Toc113547116)

[3.4. Системи базирани на правилима 9](#_Toc113547117)

[3.5. *Drools* 10](#_Toc113547118)

[4. Спецификација система 11](#_Toc113547119)

[4.1. Дијаграми случајева коришћења 13](#_Toc113547120)

[4.2. Класни дијаграм 14](#_Toc113547121)

[5. Имплементација система 17](#_Toc113547122)

[5.1. Серверска апликациja 17](#_Toc113547123)

[5.2. *Drools* систем базиран на знању 21](#_Toc113547124)

[5.2.1. Једноставна правила 21](#_Toc113547125)

[*5.2.2.* *Forward-chaining* 22](#_Toc113547126)

[*5.2.3.* *Template* 23](#_Toc113547127)

[*5.2.4.* *Backward-chaining* 24](#_Toc113547128)

[5.3. Клијентска апликација 26](#_Toc113547129)

[6. Приказ имплементираног система 27](#_Toc113547130)

[6.1. *Login* и регистрација 27](#_Toc113547131)

[6.2. Почетна страна и преглед станишта 28](#_Toc113547132)

[6.3. Додавање новог станишта 31](#_Toc113547133)

[6.4. Додавање нових антрополошких фактора 33](#_Toc113547134)

[7. Закључак 34](#_Toc113547135)

[8. Литература 35](#_Toc113547136)

[9. Биографија студента 36](#_Toc113547137)

[КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА 37](#_Toc113547138)

[KEY WORD DOCUMENTATION 38](#_Toc113547139)

# Увод

Текуница је сисар из реда глодара. То је веома љупка животиња дужине 20-ак цм, малених ушију и крупних очију, смеђесиве боје. Њен изглед карактерише поза "свећице", када животиња седи на задњим ногама и усправљеног трупа осматра околину. Ова врста распрострањена је у југоисточном делу средње Европе. У нашој земљи настањује првенствено Војводину, док је у осталом делу Србије знатно ређа.

У Војводини живи на пашњацима и утринама уз путеве, насипима и осталим необрађиваним земљиштима. Не настањује пољопривредне културе, већ их посећује само у ободним деловима. На планинама текунице живе већином на јако девастираним пашњацима са ниском зељастом вегетацијом.

Текуница је друштвена животиња и живи у колонијама у којима су суседи често веома близу. Свака јединка има под земљом своју јазбину са гнездом.

Природна станишта Војводине са комплексом живих бића која су их насељавала у прошлости су била мозаик шума, шумо-степа, степа, слатина и мочвара. Због погодности земљишта за пољопривреду човек је заузео велики део Војводине и ова природна станишта претворио у обрадиве површине. Преостале необрађене површине данас су оазе на којима живе многе ретке врсте, а текунице су једна од њих.



Слика 1.1. Текуница [1]

Због драстичног смањења бројности и низа угрожавајућих фактора, текунице се у Србији налазе на листи строго заштићених врста. Сва жива бића су у природи директно или индиректно повезана па смањење бројности једних утиче и на опстанак бројних других врста. Нестајање текуница стога није само осиромашење генетског фонда сисара, већ и угрожавање опстанка одређених птица грабљивица за које је текуница храна од виталног значаја. Међу њима је и веома редак орао крсташ, степски соко и многе друге.

Тако губитак једне карике у ланцу исхране доводи и до значајног смањења укупне биолошке разноврсности.

Због свега наведеног систем заштите природе у Србији очување и унапређење популација текуница, заједно са многим другим угроженим врстама, сврстава у приоритетне активности. Од изузетног је значаја очувати постојеће популације и обновити их (реинтродуковати) на местима са којих су нестале а постоје подаци да су их у прошлости насељавале. Такође, као додатна активност на заштити врсте планира се и интродукција на адекватним стаништима.

Свако станиште резултат је комплекса абиотичких и биотичких фактора, и наравно утицаја човека што условљава могућност живота и опстанка различитих живих бића. Разлике међу стаништима огледају се у саставу земљишта, количини падавина, надморској висини, екпозицији, вегетацији која их покрива, итд. што их, кад су текунице у питању може чинити погодним, мање погодним или потпуно непогодним за покушај реинтродукције или интродукције [1].

Систем за интродукцију и реинтродукцију текуница замишљен је као помоћ у процени адекватности одређеног подручја за њихово насељавање. Уношењем конкретних параметара о самој локацији и података сакупљених на терену, софтвер ће, на основу већ дефинисаних утицаја сваког појединачног фактора на популацију текуница, проценити да ли је предложено станиште адекватно или не за интродукцију или реинтродукцију текуница.

Циљ овог рада је да детаљно опише апликацију, њену архитектуру и начин коришћења.



Слика 1.2. Текуница [1]

# Стање у области

Процена погодности за настањивање одређене врсте може бити компликован процес и захтевати сагледање различитих фактора. Како би се олакшала анализа свих сакупљених фактора и смањила потреба за пуно стручњака из области заштите животне средине и биологије, систем базиран на знању може бити користан као испомоћ у раду.

У наредном поглављу наведени су радови у којима се говори о различитим системима који се баве заштитом животне средине и бројношћу врста на одређеним подручјима.

Рад [2] говори о избору погодних станишта за настањивање текуница на територији Румуније, као и о факторима који утичу на избор станишта. Овај рад је коришћен као основа приликом осмишљања спецификације за Систем за интродукцију и реинтродукцију текуница.

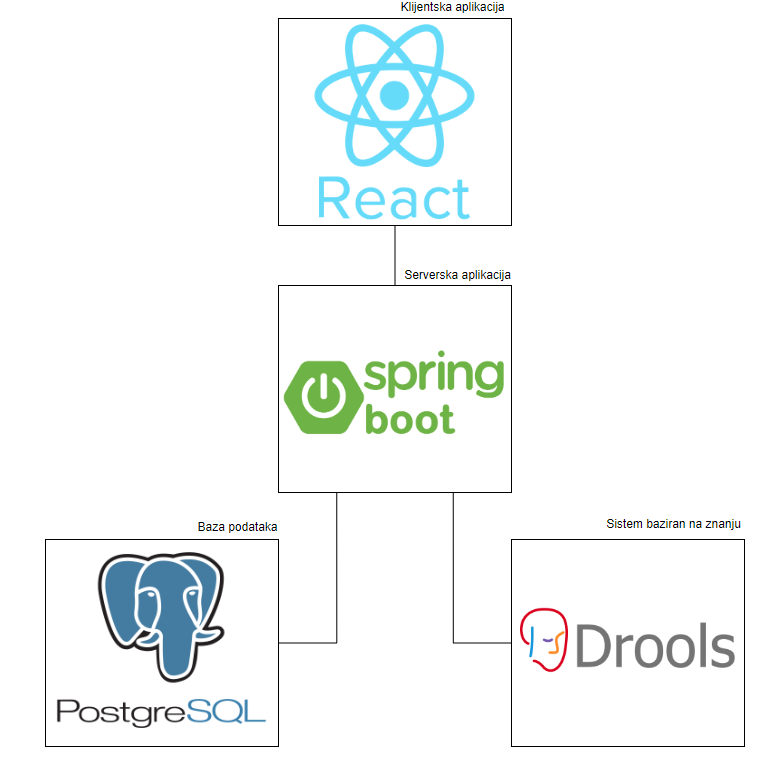
Рад [3] описује *rule-based* систем имплементиран са циљем повећања бројности рисова (*Eurasian lynx*) на територији Немачке.

Овај рад [4] говори о моделу за планирање стратегије за насељавање вилинске пите (*Fairy Pitta*) на подручу Тајвана.

Рад [5] говори о систему базираном на правилима који као задатак има да прати бројност и распрострањеност птица у Енглеској као и да предвиди распрострањеност гнездеће популације.

# Коришћене технике и технологијe

Систем се састоји из клијентске и серверске стране. За имплементацију клијентске стране, односно *frontend*-а, коришћена је библиотека *React*. Серверска страна је имплементирана у Јави, коришћењем *SpringBoot* радног окружења, и *REST* сервиса. За чување података на серверској страни коришћена је *PostgreSQL* база података. Систем базиран на правилима имплементиран је коришћењем *Drools* алата.



Слика 3.1. Архитектура апликације

Свака од наведених технологија је укратко описана у даљем тексту.

* 1. *React*

*React* [6]је *JavaScript* библиотека која се користи за креирање корисничког интерфејса. *React* код се састоји из компоненти, које су типично писане у *JSX*-у (*JavaScript Syntax Extension*) [7]. Компоненте саме чувају своје стање, и користе се за креирање сложенијих компоненти.

*React* је декларативан језик, и омогућује једноставан дизајн апликација. Библиотека сама одређује које компоненте је када потребно ажурирати приликом уноса нових измена.

*React* користи виртуално *DOM* стабло као структуру у којој се чувају компоненте. Обично *DOM* стабло представља струткуру у којој се налазе компоненте веб апликације, и при свакој измени компоненте, креира се ново *DOM* стабло које представља тренутну верзију апликације. Виртуелно *DOM* стабло омогућује измену само оне компоненте стабла у којој је стварно дошло до промене, као и компоненти које она садржи. Тиме се чувају меморијски ресурси и омогућује се брже ажурирање компоненти.

## *REST*

*REST* (*Representational State Transfer*) [8] је архитектура која описује униформан начин приступа и измене ресурса на серверу. Сам *REST* се не бави детаљима имплементације тих функционалности, већ их описује на вишем нивоу апстракције. Следе неки од основних принципа *REST*-а:

* Униформни интерфејси = сви захтеви за ресурсима требају бити истог формата. Такође, сваки ресурс треба да има свој јединствени *URI* (*Uniform Resource Identifier*) преко ког му се може приступити.
* Независност клијентске и серверске стране = клијентска и серверска апликација требају бити потпуно независне једна од друге. Клијенту је потребно само да зна *URI*-je преко којих тражи податке од сервера, док сервер на основу пристиглог *URI*-ja враћа одговарајуће податке.
* *Stateless* комуникација = сваки захтев за подацима који клијент пошаље треба да буде независтан од осталих. Он мора садржати све информације неопходне да сервер врати тачно оне податке који су тражени. Сервер не мора, и не треба, да памти податке о клијенту, као ни о предходним захтевима који су пристигли.

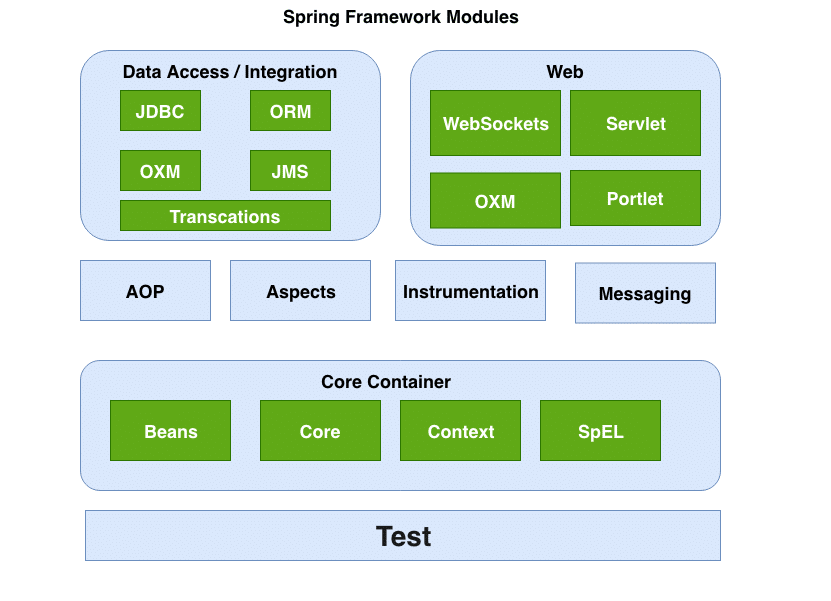
У систему је коришћен *REST* који се ослања на *HTTP* (*Hypertext transfer protocol*) [9]. Главне операције за приступање ресурсима које су коришћене су и стандардне *HTTP* методе: *GET*, *POST*, *PUT* и *DELETE*.

## *Spring Boot*

*SpringBoot [10]* је радно окружење коришћено за имплементацију *REST* сервиса. Главне карактеристике *Spring*-a [11] су да омогућује креирање Јава апликација на једноставнији начин, управља инфраструктуром за изградњу апликација тако да програмер може да се фокусира на доменске проблеме, и омогућује брзу и ефикасну реализацију стандардних функционалности.

*Spring* је подељен на модуле који садрже подршке за развој различитих аспеката апликација. *Core Spring Container* је главни модул и задужен је за управљање животним циклусом објеката у пројекту. Поред овог, постоје и модули за управљање подацима, веб модул, модул за подршку аспектно-оријентисаног програмирања, и многи други.

На следећој слици приказани су модули *Spring* радног окружења.



Слика 3.2. *Spring* модули [12]

*SpringBoot* је један од *Spring* пројеката, и представља радно окружење за *Spring* радно окружење. Другим речима, *SpringBoot* поједностављује корићење *Spring*-a и обавља посао конфигурисања *Spring*-a за корисника.

* 1. Системи базирани на правилима

Компјутерски систем који обавља задатке сличне онима које стручњаци неке области обављају назива се експертски систем. Експертски систем симулира рад стручњака одговарајућег домена, користећи људску експертизу репрезентовану у облику правила.

Правила служе да се представе релације у подацима, као и стратегије за долазак до крајњег закључка.

Правило је продукционо уколико је наведено у *IF-THEN* форми (IF „услов“ THEN „акција“). Продукциона правила која енкапсулирају знање стручњака користе се за доношење закључака над конкретним подацима у систему.

Уколико експертски систем садржи продукциона правила, он се зове „Систем базиран на знању“.

Подаци у контексту система базираних на знању зову се чињенице. Оне представљају стање система, и не садрже никакве информације о потенцијалним акцијама које треба предузети.

Све чињенице које постоје у систему чувају се у бази знања.

Додавањем нове чињенице у систем, активирају се сва правила и уколико чињеница задовољи услов неког правила, његова акција ће се извршити.

Правила у систему могу се уланчавати, и постоје две врсте уланчавања: уланчавање унапред (*forward chaining*) и уланчавање уназад (*backward chaining*).

Уланчавање унапред је вођено подацима. Нове чињенице које изађу из једног правила, могу активирати неко друго правила и тиме добијамо уланчана правила. Чињеница која изађе из правила и не успе да активира ни једно друго, представља излаз из система. У уланчавању унапред не може се увек знати шта ће бити резултат операција, као ни да ли ће се до њега уопште доћи.

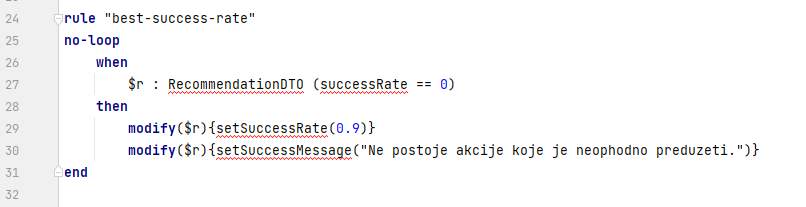
Уланчавање уназад са друге стране је вођено циљем. У овој ситуацији желимо да докажемо да нека хипотеза важи, тако што ћемо кренути од ње и ићи уназад кроз правила како би дошли до основне чињенице коју требамо имати у систему.

* 1. *Drools*

*Drools* [13] је алат коришћен за дефинисање и извршавање правила у систему. Он обезбеђује централизовано знање, односно раздваја саму апликацију од њене експертске логике.

Правила написана помоћу *Drools Rule Engine*-a су читљива и разумљива и не експертским корисницима, односно и онима који се не баве програмирањем.

На следећој слици (Слика 3.3) може се видети једноставно *Drools* правило, које се активира кад се у бази знања појави чињеница *RecommendationDTO* која задовољава услов да је вредност параметра *successRate* = 0, и мења вредност атрибута *successRate* и *successMessage*.



Слика 3.3. Једноставно *Drools* правило

Свако правило написано у *Drools*-у састоји се од леве и десне стране. Лева страна, скраћено *LHS* (*left-hand side*), представља услов, који чињеница мора задовољити да би се ушло у правило. Десна страна односно *RHS* (*right-hand side*), представља акцију која се извршава уколико је услов задовољен. Поред леве и десне стране, свако правило има почетак и крај, име, и опционе атрибуте. Један од атрибута, који се може видети и у самом пројекту је *no-loop* који означава да правило неће активирати само себе, односно да ако чињеница кад изађе из правила и даље задовољава његову леву страну, правило се неће опет покренути.

# Спецификација система

Систем за интродукцију и реинтродукцију текуница је веб апликација која за намену има да анализира описано станиште и да да процену да ли је оно погодно за настањивање текуница, као и да предложи које мере би могле да се предузму како би станиште било погодније.

За коришћење апликације није неопходно, али је пожељно имати основно знање из области биологије и географије.

Станиште, или локалитет, је у систему описано са следећим атрибутима:

* име станишта
* природни фактори
* антрополошки (људски) фактори
* оцена станишта

Природни фактори су непроменљиве особине локалитета и оне највише утичу на погодност самог станишта за настањивање текуница. У природне факторе спадају тип станишта (степа, пашњак, шума,..), надморска висина, средња јулска температура, нагиб терена, експозиција и учесталост падавина и плављења терена.

Антрополошки или људски фактори су углавном акције које људи директно или индиректно спроводе, које могу негативно утицати на погодност станишта. У ову групу спадају близина саобраћајница, испаша, крађа траве, тровање и многи други. Антрополошки фактори се могу звати и променљиви, јер се на њих може утицати. За разлику од природних фактора они не утичу на оцену локалитета, али се на основу њих кориснику препоручују акције које може извршити са циљем да унапреди станиште за живот текуница.

Како се систем може користити и за праћење напретка акција над одређеним локалитетом, антрополошке факторе је могуће уносити више пута за исто станиште.

Приликом давања препорука на основу антрополошких фактора, систем такође рачуна и проценат колика је вероватноћа да ће, ако се све предвиђене мере предузму, текунице успети да се настане на локалитету. Та вредност је обрнуто сразмерна броју акција које су препоручене кориснику да их изврши.

Оцена станишта може имати једну од следећих шест вредности:

* Оптимално станиште
* Субоптимално станиште
* Средње добро станиште
* Неадекватно станиште
* Неповољно станиште
* Нема лабеле

Последњу вредност оцена има само приликом креирања станишта, док се станишту не додели једна од осталих пет оцена.

Уколико се локалитету додели оцена “Неповољно станиште”, оно се сматра јако непогодним за настањивање текуница, и из тог разлога корисник неће моћи да додаје нове људске факторе. Кориснику се такође неће исписати препоруке за побољшање станишта, јер је оно по својим непроменљивим особинама добило ту ознаку.

* 1. Дијаграми случајева коришћења

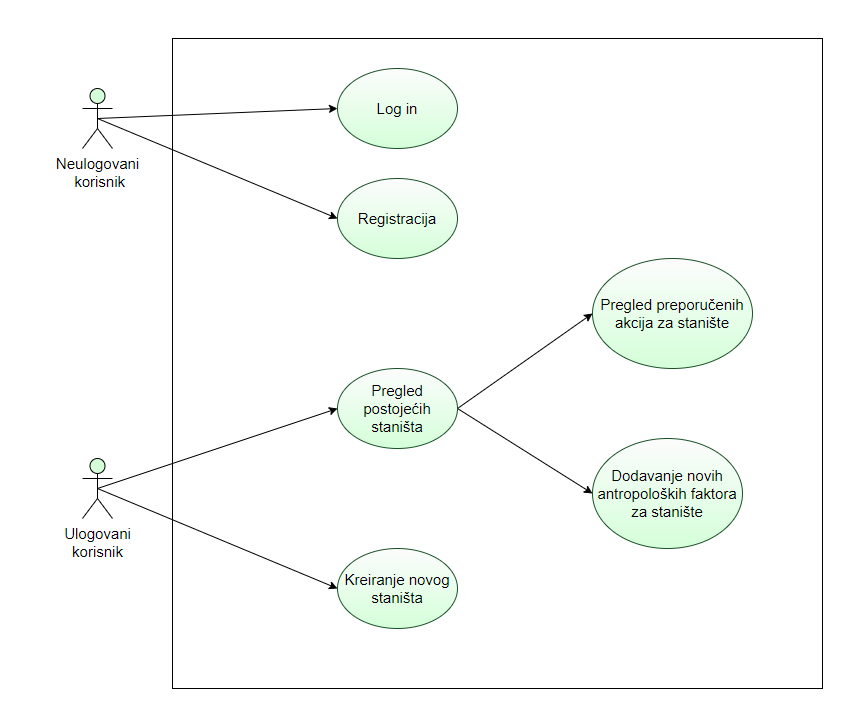
За коришћење апликације потребно је имати кориснички налог. Уколико он не постоји, може се креирати на почетној страни апликације и потом користити.

Улоговани корисник има могућност да креира ново станиште, као и да прегледа сва станишта која је претходно додао.

Приликом додавања новог станишта у систем, корисник попуњава форму у коју уноси све претходно наведене особине станишта, осим оцене. Систем потом одређује оцену станишта, и проналази одговарајуће акције које ће кориснику бити препоручене и приказује их кориснику.

Уколико корисник одабере приказ већ постојећег станишта, приказаће му се природни фактори, оцена станишта као и листа свих антрополошких фактора који су додати за то станиште. Корисник може да унесе нове вредности, као и да види препоруке за претходно креиране антрополошке факторе.

На следећој слици (Слика 4.1) се може видети графички приказ предходно описаних случајева коришћења.



Слика 4.1. Дијаграм тока коришћења апликације

## Класни дијаграм

Апликација садржи следеће класе:

* *User*
* *Authority*
* *Habitat*
* *NaturalFactors*
* *AnthropologicalFactors*
* *AnthropologicalFactorLevelAndDescription*
* *Option*

Класе User и Authority служе за аутентификацију корисника приликом логовања на систем.

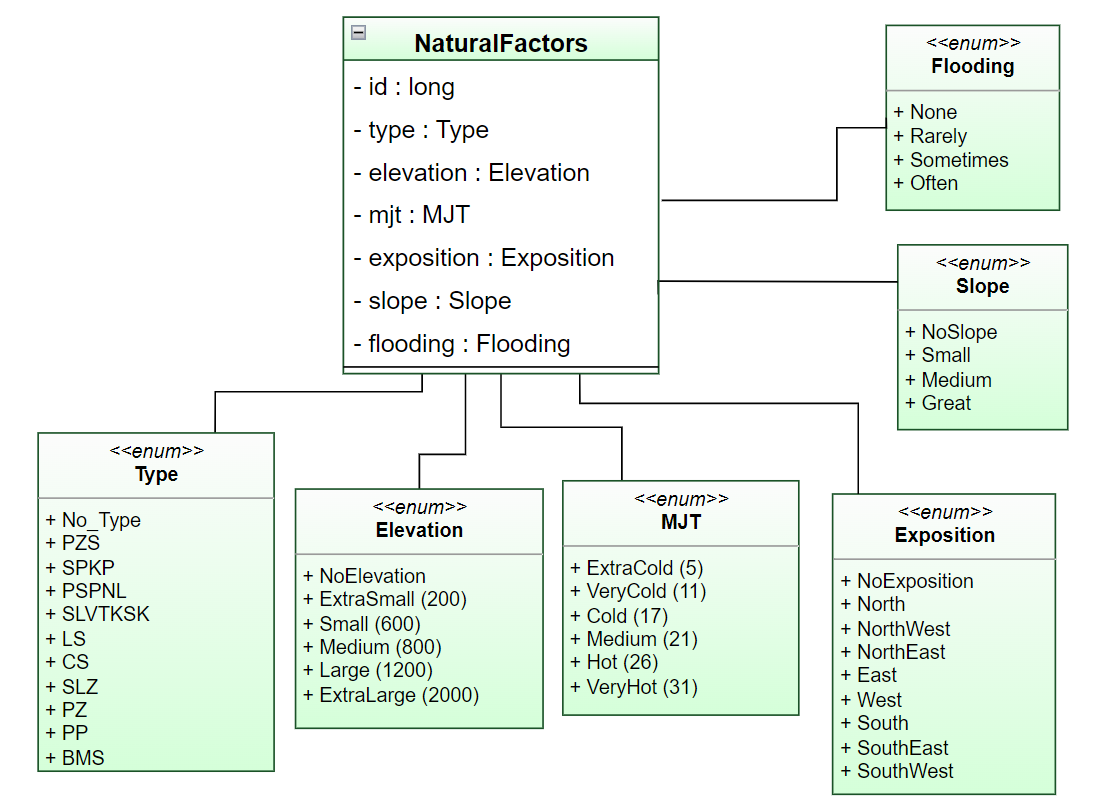
Класа *Habitat* представља станиште, и садржи све предходно наведене атрибуте станишта.

Природни и антрополошки фактори су дефинисани класама *NaturalFactors* i *AnthropologicalFactors*, које садрже атрибуте за све појединачне вредности фактора.

Вредности атрибута у *NaturalFactors* класи су представљене као вредности из следећих енумерација:

* *Type* (тип станишта)
* *Elevation* (надморска висина)
* *MJT* (средња јулска температура)
* *Exposition* (експозиција)
* *Slope* (нагиб терена)
* *Flooding* (учесталост падавина и плављење терена)

На следећој слици (Слика 4.2) налази се класни дијаграм за класу *NaturalFactors* и све енумерације које су са њом повезане.



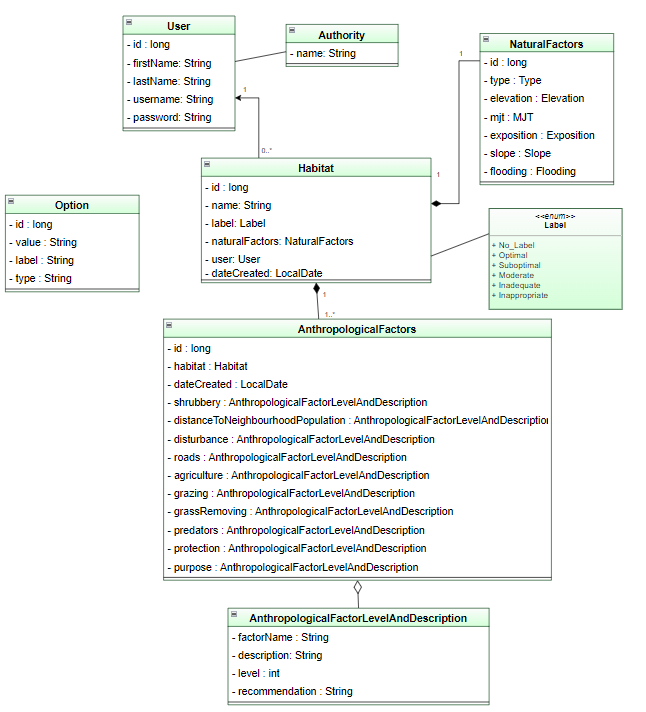
Слика 4.2. Класни дијаграм за природне факторе

Вредности појединачних антрополошких фактора у класи *AnthropologicalFactor* представљене су помоћу класе *AnthropologicalFactorLevelAndDescription*. Ова класа за атрибуте има назив фактора, текстуални опис (који се кориснику приказује у форми приликом додавања антрополошких фактора) и бројну вредност од 0 до 10 која означава колико је фактор неповољан (0 = стање веома повољно, 10 = стање веома неповољно).

Класа *Option* служи за чување свих вредности које корисник може одабрати приликом креирања станишта, као и за размену тих вредности између клијентске и серверске апликације.

Поред претходно описаних енумерација, постоји и енумерација *Label*, која служи за представљање оцене станишта.

На следећој слици (Слика 4.3) налази се класни дијаграм који приказује све класе у систему.



Слика 4.3. Класни дијаграм целе апликације

# Имплементација система

У следећем поглављу детаљније су описане серверска и клијентска страна апликације као и *Drools* систем базиран на знању.

Изворни код апликације се може наћи на следећем линку: <https://github.com/isidorasavic/tekunice>.

* 1. Серверска апликациja

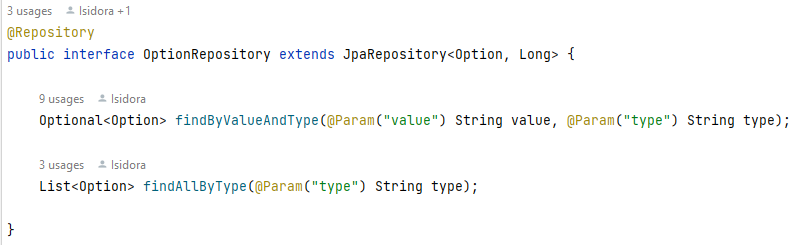
Серверска апликације као главни задатак има да прима захтеве од клијентске апликације, и враћа одговарајуће одговоре. Она такође комуницира са *Drools* апликацијом.

Серверска апликација је организована у следеће фолдере:

* *configuration*
* *controller*
* *dto*
* *exception*
* *facts*
* *repository*
* *security*
* *service*

У директоријуму *facts* налазе се све класе и енумерације које су описане у претходном поглављу.

Репозиторијуми служе за комуникацију апликације са базом података. Сви се налазе у фолдеру *repository*, и сви имплементирају *JpaRepository* [14] интерфејс. Методе интерфејса које су коришћене су све *Query* методе (*Query method*) [14]. *Query* методе омогућују једноставније писање метода за рад са базом података, без потребе да се пишу тачни упити. Неопходно је методи проследити само одговарајуће параметре, и декларисати је у складу са тим шта треба да обави, док ће сам посао дефинисања методе обавити радно окружење за нас. На следећој слици (Слика 5.1) налази се пример репозиторијума који служи за рад са корисницима.



Слика 5.1. Пример репозиторијума за рад са корисницима

Сервиси су класе које обављају све операције над подацима у систему и налазе се у фолдеру *sevices*. Постоје четири сервиса у систему:

* *CustomUserDetailsService*
* *NaturalFactorsService*
* *AnthropologicalFactorsService*
* *HabitatService*

*CustomUserService* извршава све функције неопходне за логовање корисника на систем, као и за регистрацију.

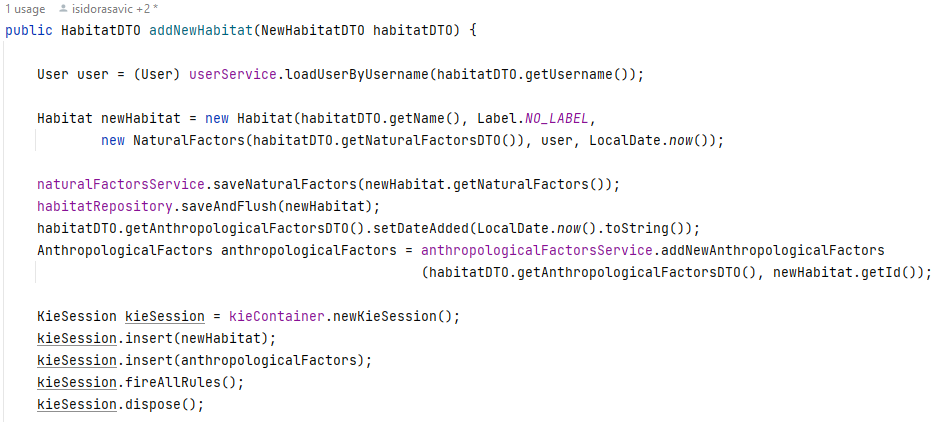
*NaturalFactorService* садржи методе везане за чување природних фактора станишта у бази података, као и за добављања опција за креирање нових природних фактора. У наредном поглављу ће ово бити детаљније описано.

*AnthropologicalFactorsService* је задужен за рад са антрополошким факторима, њиховим додавањем и добављањем из базе података, као и за састављање препорука акција за корисника. Како се препоруке добијају из антрополошких фактора ће такође бити детаљније описано у следећем поглављу.

*HabitatService* ради са стаништима у систему. Овај сервис садржи методе за додавање новог станишта у систем. Приликом додавања новог станишта, серверска апликација комуницира са *Drools* системом, како би се одредила оцена станишта.

Сви сервиси, осим *CusomUserService*, су повезани са *Drools* системом. Наиме на почетку рада апликације креиран је *bean* *KieContainer*. Сервиси су повезани са овим *bean*-ом помоћу механизма *dependency injection* [15]. *KieContainer* се користи за генерисање *KieSession*-a која слузи за покретање правила из система. Сваки пут кад на серверску апликацију стигне захтев који захтева коришћење система базираног на знању, креира се сесија, у њу се убацују неопходни подаци, и потом се покрећу сва правила позивом методе *fireAllRules()*. Пошто су се активирала сва правила чије услове су предходно убачене чињенице задовољиле, гаси се и сесија.

На следећој слици (Слика 5.2) види се исечак из кода, у ком се додаје нови локалитет у систем, креира се сесија и покрећу се правила, како би се одредила оцена локалитета.



Слика 5.2. Пример сервиса за станишта

Контролери су задужени за примање *HTTP* захтева и слање одговарајућих одговора. На основу самог захтева, позива се одговарајућа метода сервиса која је задужена за извршавање потребних операција како би се клијенту послао тражени ресурс.

На следећој слици (Слика 5.3) се може видети пример контролера за станишта, који на основу захтева позива *HabitatService* класу и враћа одговарајући одговор.



Слика 5.3. Пример контролера за станишта

За потребе управљања грешкама до којих може доћи на серверској страни, креирано је неколико класа у *exception* фолдеру, које представљају различите грешке до којих може доћи. Креирана је и *RestExceptionHandler* класа која на сваки захтев који резултује грешком одговара са објектом који садржи време у ком је дошло до грешке, поруку о грешци и одговарајући *HttpStatus* код [16].

Директоријум *dto* садржи *DTO* класе за све класе у систему, и служи за размену података између клијентске и серверске апликације.

* 1. *Drools* систем базиран на знању

Систем базиран на знању имплементиран је помоћу *Drools*-a, и он се налази у засебном фолдеру “*drools-spring-kjar*”.

Сви фајлови који садрже правила, са екстензијом *.drl*, груписани су у *resources* директоријум пројекта. Правила су организована по функционалностима које имплементирају. У овој апликацији имплементиране су следеће компоненте *Drools* језика:

* једноставна правила
* *forward-chaining*
* *template*
* *backward-chaining*

У даљем тексту описано је како су све наведене компоненте имплементиране.

### Једноставна правила

Једноставна правила су класична IF-THEN правила која на основу атрибута објеката из радне меморије доносе закључке. Нису уланчана и резултат једног правила не активира неко друго правило.

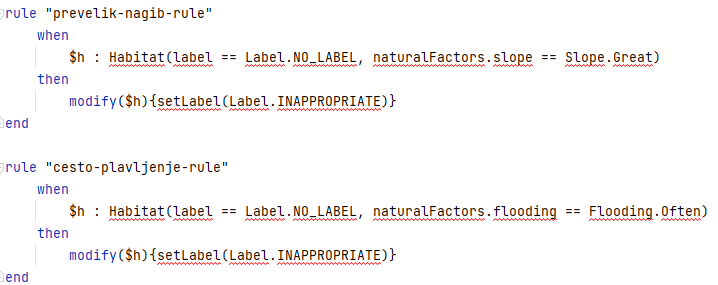
У апликацији, једноставна правила су коришћена за одређивање оцене станишта у ситуацијама, где само један или два природна фактора могу одредити његов квалитет.

Једна од тих ситуација јесте кад је нагиб станишта већи од 40%, или пак кад нагиба нема (0%) а плављење станишта је често. Станишта која имају ове особине одмах се могу прогласити непогодним, без обзира на све остале факторе.

Други пример може бити станиште чији је тип „Панонска заслањена степа” а експозиција „Северо-исток”. Овакво станиште је сигурно онда субоптимално.

У свим једноставним правилима за одређивање оцене станишта може се уочити и услов да је оцена станишта „NO\_LABEL”. На крају правила уколико је услов задовољен, станиште добија неку другу вредност оцене, и тиме је спречено уланчавање правила као и бесконачна рекурзија правила.

На следећој слици (Слика 5.4) види се пример једноставног правила из апликације. Прво правило се активира када се у радној меморији нађе станиште које нема додељену оцену, и има велик нагиб. Друго правило се активира уколико постоји станиште које нема оцену и падавине и плављење су чести. И прво и друго правило ако се активирају, постављају вредност оцене станишта на “Неповољно станиште“.



Слика 5.4. Једноставна правила за одређивање неповољних станишта

### *Forward-chaining*

Као што је описано у досадашњим поглављима, *forward-chaining* означава уланчана правила, тако да излаз једног задовољава услове за улазак у следеће правило.

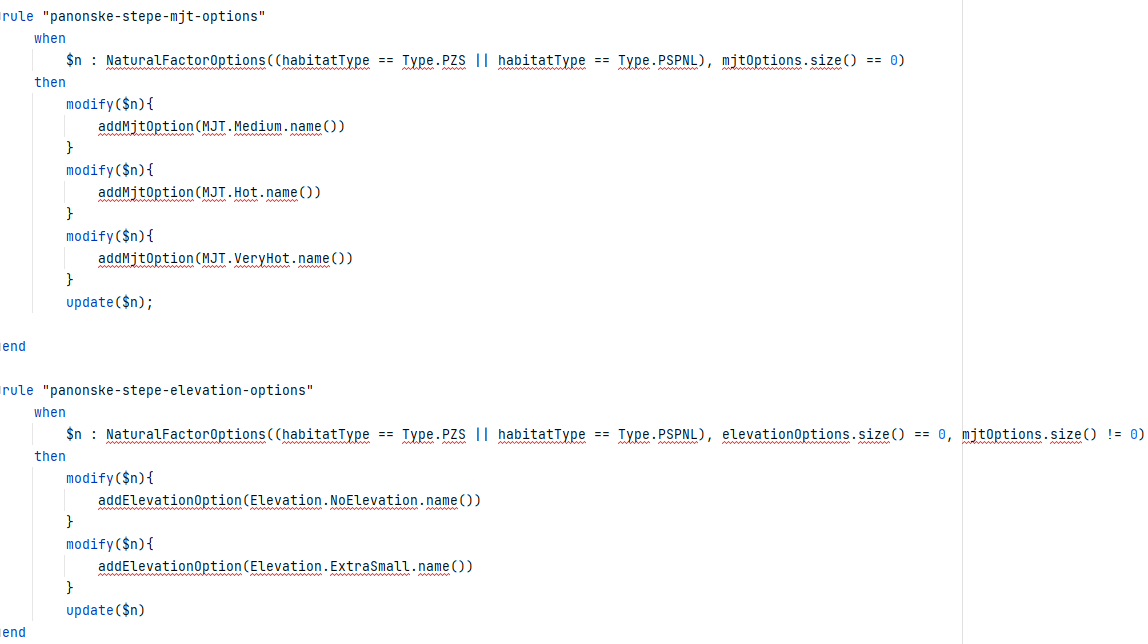
У пројекту *forward-chaining* користи се за проналазак могућих опција приликом уношења природних фактора станишта.

Наиме, уколико је одређен тип станишта, он се по својим географским особинама не може наћи на било којој надморској висини, или имати било коју средњу јулску температуру. Примера ради уколико је станиште степа, оно се налази сигурно на висини до 600м и средња јулска температура је већа од 15℃.

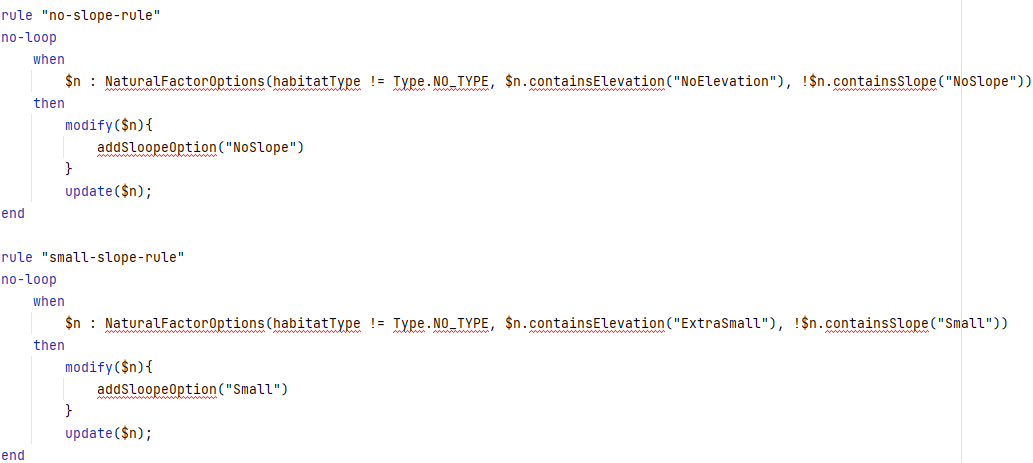
Опције за креирање природних фактора представљају појединачне вредности свих претходно наведених енумерација.

Приликом креирања новог станишта корисник прво бира тип станишта. На основу типа, и уколико опције за средњу јулску температуру нису додате, оне се додају у листу. Потом се на основу типа, и ако су опције за средњу јулску температуру додате, а за надморску висину нису, опције за надморску висину се додају. На крају уколико су листе опција и за средњу јулску температуру и за надморску висину попуњене, додају се и опције за нагиб терена. Остали природни фактори нису везани за тип, ни за друге природне факторе, те су кориснику све опције сваки пут доступне.

На слици 5.5 налазе се орва два, од три уланчана правила. Прво правило () се активира уколико се у радој меморији нађе објекат класе *NaturalFactorOptions* који за тип станишта има вредност PZS или PSPNL из енумерације Type, и нема додељене опције за средњу јулску температуру. Уколико се задовољи лева страна овог правила, објекту ће бити додељене потенцијалне вредности за средњу јулску температуру. Друго правило (Слика 5.5) се активира уколико се у меморији нађе објекат исте класе, који има предходно наведену вредност за тип станишта, има попуњену листу потенцијалних вредности за средњу јулску температуру и нема додељене вредности за надморску висину. Ако је задовољена лева страна овог правила, објекту се додељују потенцијалне вредности за надморску висину. Последња правила у уланчавању унапред налазе се на слици (Слика 5.6.). Она се активирају уколико су објекту додељени тип, опције ѕа средњу јулску температуру као и опције за надморску висину. У десној страни овог правила додају се опције за нагиб терена станишта.



Слика 5.5. Прва два правила за *forward-chaining*



Слика 5.6. Треће правило за *forward-chaining*

### *Template*

У *Drools*-у темплејти (eng. *template*) се користе за генерисање правила. Они представљају шаблон у који кад се убаце конкретни подаци, складиштени у форми табеле или листе, добију се конкретна правила.

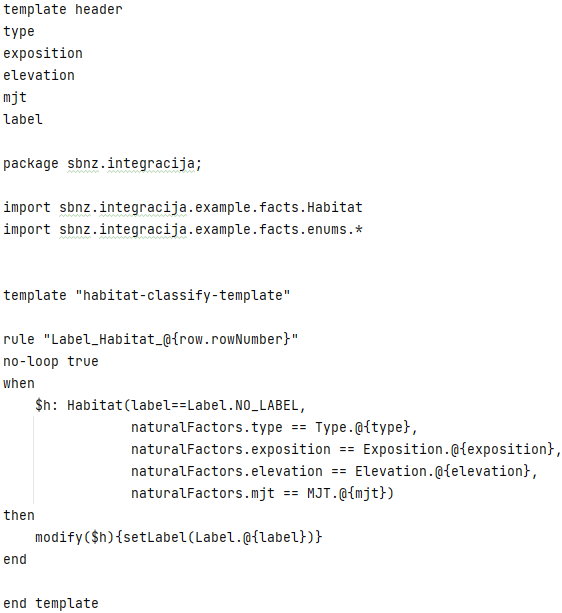
Темплејти се користе у ситуацијама кад и мале промене вредности у радној меморији могу довести до драстичних промена у очекиваном излазу.

У овом пројекту, темплејти су коришћени за генерисање већине правила за класификацију станишта. Наиме кад се одређује оцена локалитета, углавном је битна тачна комбинација типа станишта, средње јулске температуре, надморске висине и експозиције. Из тог разлога једноставније је било генерисати сва та правила помоћу шаблона.

Темплејт генерише правила која у левој страни имају услове да станиште нема већ додељену лабели, и да сви претходно наведени непроменљиви фактори имају тачно одређене вредности.

Сам темплејт је дефинисан у *habitat-classify-template.drt* фајлу, док су сви подаци на основу којих се генеришу правила складиштени у листи, у *generateRules()* методи *HabitatService* класе.

На следећој слици (Слика 5.7) се налази код којим је декларисан претходно описан темплејт.



Слика 5.7. Пример темплејта (*template*)

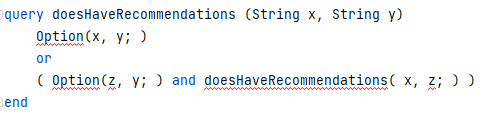
### *Backward-chaining*

*Backward-chaining* се у систему користи за проналазак препорука акција, као и за одређивање успеха акција.

Сви антрополошки фактори које је могуће одабрати у систему уз себе имају сачувану кратку препоруку која ће се потенцијално исписати кориснику.

Када са клијентске стране стигне захтев за препорукама за одређене антрополошке факторе, прво се сви фактори смештају у једну листу заједно са препорукама које уз њих стоје.

Потом се позива *query*, који проналази све препоруке облика „Није потребно предузети никакве акције.”. Те препоруке стоје уз оне факторе чија је вредност 0, односно који су задовољавајући и није их потребно мењати. На следећој слици (Слика 5.8) види се горе споменути *query*.



Слика 5.8. Query за *backward-chaining*

Уколико се пронађе свих десет таквих фактора, систем кориснику неће препоручити ништа да измени, само ће му приказати поруку о квалитету антрополошких фактора.

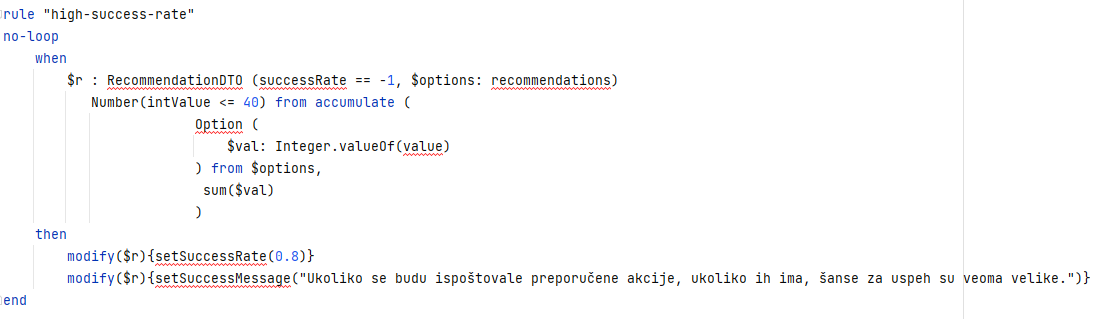
Овако је имплементирана претрага уназад, тако да се креће од претпоставке да су људски фактори сви одлични и да акције неће бити потребне, али се може десити да се та претпоставка и не докаже.

Уколико се нађе барем један фактор који нема вредност 0, покрећу се правила која рачунају вероватноћу да препоруке за све факторе, уколико се изврше, успеју.

Ова правила пролазе редом кроз све антрополошке факторе и сабирају њихове вредности, користећи *Drools* функцију *accumulate*. Уколико је добијена сума мања од 40, систем закључује да акција које треба предузети нема пуно, и поставља вероватноћу успеха на вредност 80%. Ако је сума у интервалу од 40 до 60, вероватноћа успеха ће бити 50%, док ако је сума већа од 60, вероватноћа пада на 20%.

Поред вероватноће успеха, систем поставља вредност и поруке о успеху, која кориснику укратко објашњава значење бројне вредности.

На следећој слици (Слика 5.9) налази се приказ функције која рачуна суму и уколико је она мања од 40 поставља одговарајуће вредности претходно споменутих параметара.



Слика 5.9. Правило за одређивање вероватноће успеха акција

* 1. Клијентска апликација

Клијентска страна развијана је у *React*-у. Коренска компонента апликације је *App* и у њој се налазе све остале компоненте апликације.

Апликација је организована у пет основних страница за:

* пријаву на систем
* регистрацију
* преглед свих станишта
* додавање новог станишта
* додавање нових антрополошких фактора за постојеће станиште

У позиву *return* наредбе сваке компоненте налазе се *html* компоненте које се кориснику приказују. За операције са податцима на клијентској страни коришћен је *custom hooks* механизам [17].

Захтеви се на сервер шаљу користећи *axios* библиотеку [18].

На слици 5.10 види се исечак из кода компоненте креирање новог станишта.



Слика 5.10. Пример компоненте за додавање природних станишта

# Приказ имплементираног система

У наредном поглављу биће приказан рад система из угла корисника, као и све функционалности које су имплементиране.

ТОДО

## Пријава и регистрација

Приликом покретања апликације, отвара се почетни прозор на ком корисник може да се улогује у систем или да креира нови налог.



Слика 6.1. Страница за login

Кликом на дугме у пољу „Lozinka” корисник може да прикаже лозинку коју је унео, или да је врати у скривени формат.

Кликом на линк „KREIRAJ NALOG” отвара се нови прозор на ком корисник може да креира свој налог на систему.

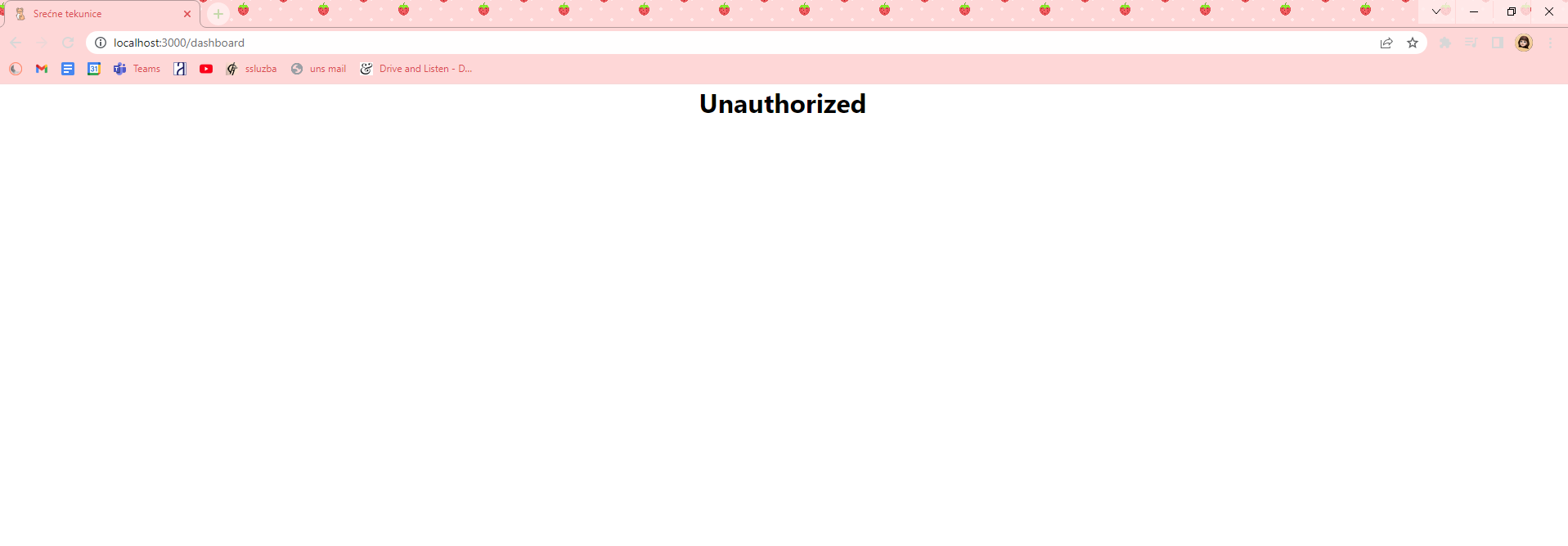


Слика 6.2. Страница за регистрацију

На овом прозору неопходно је да корисник унесе корисничко име, и лозинку у два поља. Уколико се лозинке не поклапају, биће исписана порука, и дугме „SUBMIT ” ће бити блокирано.

Кад корисник успешно пошаље своје податке, уколико корисник са истим корисничким именом не постоји већ у систему, нови налог ће бити креиран, и корисник ће бити враћен на “логин” страницу. Уколико дође до неке грешке и налог не буде успешно креиран, кориснику ће бити исписана одговарајућа порука. Приликом клика на „LOG OUT” дугме, корисник се излогује из система, и враћа на почетну *login* страницу.

Уколико неулогован корисник покуша да приступи некој од осталих страница апликације преко *URL*-a, биће му приказан следећи прозор (Слика 6.3).



Слика 6.3. Приказ екрана приликом неауторизованог приступа страницама

## Почетна страна и преглед станишта

Пошто је корисник улогован на систем, отвара се почетни прозор на ком се налазе сва његова станишта (Слика 6.4).



Слика 6.4. Почетна страница

Са леве стране се налази листа свих постојећих локалитета („Нерадин” и „Пашњак”). На главном делу екрана се налази приказ тренутног станишта, и његових природних фактора као и оцене станишта уз кратак опис саме ознаке.

Поред природних фактора налази се листа антрополошких фактора креираних за одабрано станиште, као и датум када су креирани.

Кликом на појединачни таб приказују се антрополошки фактори за тај датум, као и препоруке система. Поред самих препорука, приказана је и вероватноћа за успех настањивања текуница, уколико се спроведу све акције (Слика 6.5).



Слика 6.5. Приказ антрополошких фактора и препорука

Уколико су сви антрополошки фактори добри, систем не препоручује никакве акције (Слика 6.6).



Слика 6.6. Приказ антрополошких фактора без препоруке

Као што је речено, систем сваком станишту даје једну од пет могућих оцена. За сваку оцену постоји одређени опис оцене. Слика 6.4 приказује опис оцене оптималног станишта.

На следећој слици налазе се описи осталих оцена станишта (Слика 6.7).



Слика 6.7. Описи осталих оцена стнаништа

Уколико је станиште неповољно, кориснику се неће приказати акције које би побољшале станиште. Уместо акција биће приказана одговарајућа порука (Слика 6.8).



Слика 6.8. Препоруке за неповољно станиште

Уколико корисник нема још увек ни једно станиште у апликацији, почетни екран изгледа као слици .

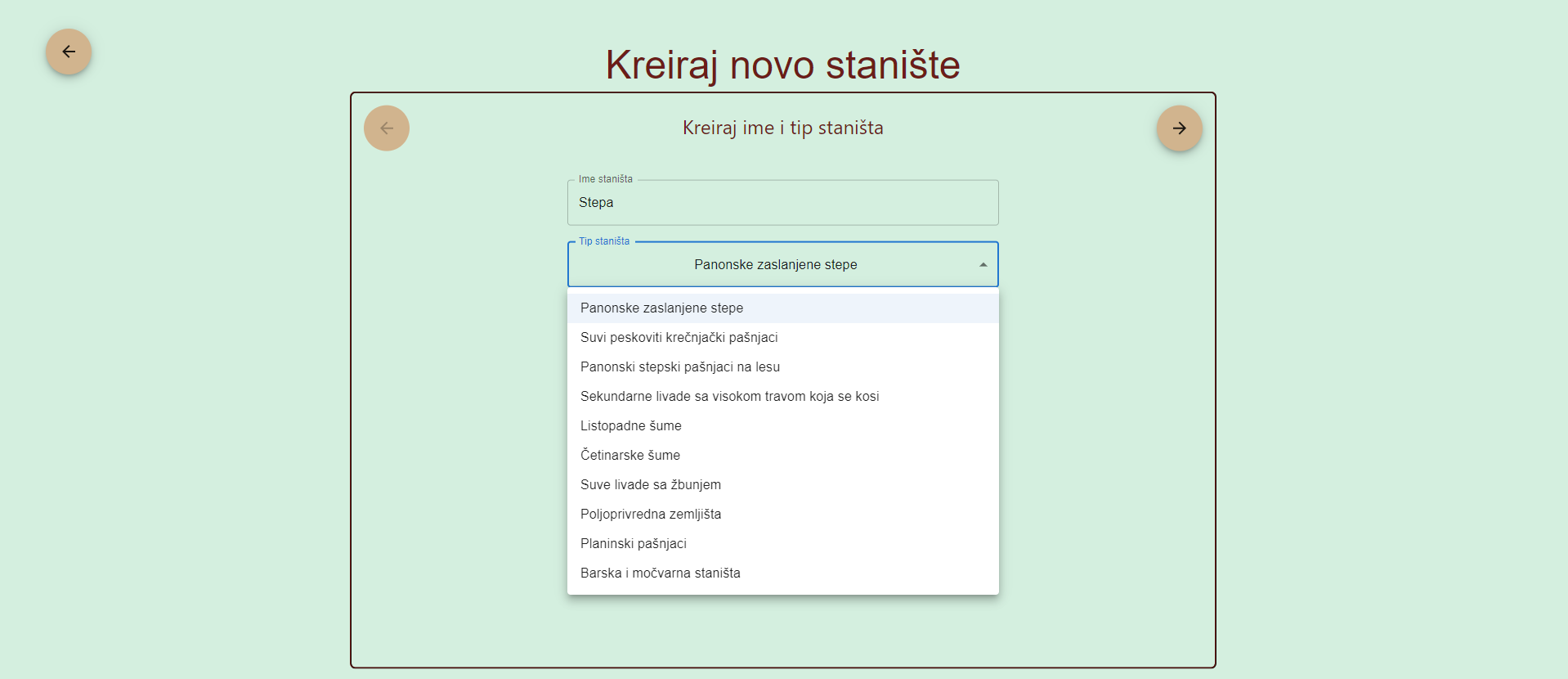


Слика 6.9. Почетна страна за корисника без станишта

## Додавање новог станишта

Било да корисник већ има станишта у систему или не, може додати ново станиште кликом на дугме „DODAJ NOVO STANIŠTE”.

Тада се отвара прозор за унос имена и типа станишта (Слика 6.10).



Слика 6.10. Форма за креирање новог станишта

У горњем левом углу се налази стрелица која враћа корисника на почетни екран, и брише унесене податке о новом станишту.

Кликом на две стрелице на средини екрана корисник може да се креће кроз прозоре за додавање имена и типа станишта, природних фактора и антрополошких фактора.

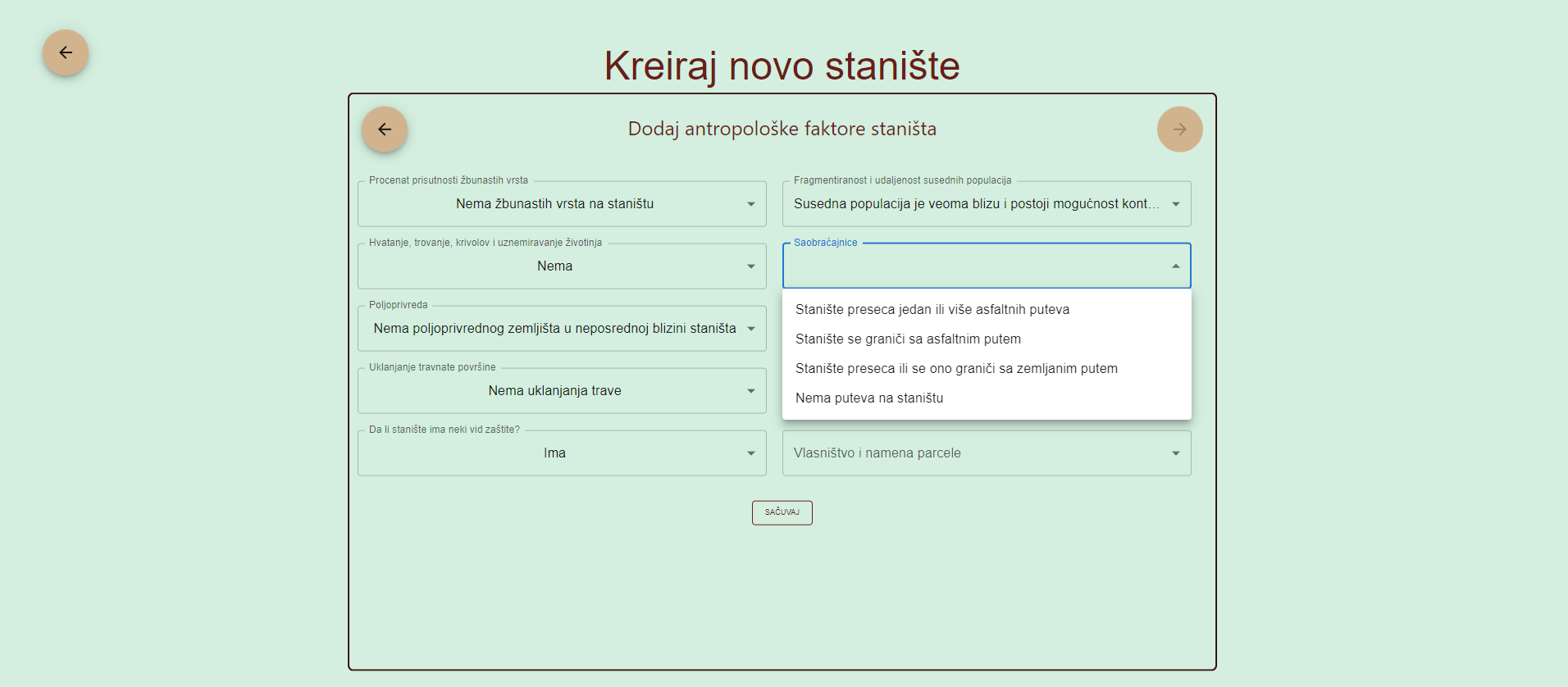
Пошто је одабран тип станишта, сервису се шаље захтев за опцијама за природне факторе (Слика 6.11).



Слика 6.11. Форма за додавање природних фактора

Уколико корисник не одабере ни један тип, на овој страници неће моћи да одабере ни једну вредност за надморску висину, средњу јулску температуру и нагиб, како систем није прошао кроз правила и не зна које тачно вредности за ове факторе станиште може да има.

Последњи прозор при креирању станишта садржи форму за унос људских фактора (Слика 6.12).



Слика 6.12. Форма за додавање антрополошких фактора

Пошто корисник попуни све три форме, кликом на дугме „SAČUVAJ” ново станиште се чува у систему, и корисник се враћа на почетну страну. Уколико је неко од поља форме остало празно у моменту кад корисник кликне дугме „SAČUVAJ” биће му исписана порука да је неопходно да попуни сва предвиђена поља.

На почетном екрану кад је одабрано једно станиште, уколико оно није неповољно, поред листе антрополошких фактора налази се и дугме „+” које служи за додавање нових антрополошких фактора за то станиште.

Прозор који се тад отвара је исти као на слици 6.12. Кликом на „SACUVAJ” дугме на овом прозору додају се нови људски фактори у систем, и приказују у свом станишту, као и нове препоруке за акције. Датум кад су антрополошки фактори додати, као и датум када је станиште креирано су тренутни датуми када су унесене измене, и корисник их не може ручно мењати.

# Закључак

Овај рад детаљно описује рад апликације за интродукцију и реинтродукцију текуница. Описане су технологије које су за њен развој коришћене, као и сама структура апликације. Апликација се састоји из три подсистема: серверска апликација, клијентска апликација и *Drools* систем базиран на знању. Серверска апликација имплементирана је у *SpringBoot* радном окружењу, клијентска у *React*-у, а систем базиран на знању у *Drools*-у.

Описана је спецификација система, и које све функционалности систем треба да извршава. Детаљно је анализиран код сва три дела апликације и приказани су исечци из стварног кода.

У последњем поглављу је приказан и рад саме апликације из угла корисника, као и сви прозори које корисник може видети приликом коришћења апликације.

Апликација би се даље могла унапредити додавањем још функционалности везаних за праћење бројности текуница на локалитету, и уколико се примети да тај број расте/опада, кориснику би се препоручиле неке друге акције.

# Литература

1. <http://www.naturefg.com/pages/c-animals/spermophilus%20citellus.htm> - преузето у септембру 2022
2. Zaharia, G., Petrencu, L., & BALTAG, E. Ş. (2016). Site selection of European ground squirrels (Spermophilus citellus) in Eastern Romania and how they are influenced by climate, relief, and vegetation. *Turkish Journal of Zoology*, *40*(6), 917-924. - преузето у септембру 2022
3. Schadt, S., Knauer, F., Kaczensky, P., Revilla, E., Wiegand, T., & Trepl, L. (2002). Rule‐based assessment of suitable habitat and patch connectivity for the Eurasian lynx. *Ecological applications*, *12*(5), 1469-1483. - преузето у септембру 2022
4. A rule-based species predictive model for the vulnerable Fairy Pitta (Pitta nympha) in Taiwan. *Taiwania*, 2009, 54.1: 28-36. - преузето у септембру 2022
5. Tucker, K., Rushton, S. P., Sanderson, R. A., Martin, E. B., & Blaiklock, J. (1997). Modelling bird distributions—a combined GIS and Bayesian rule-based approach. *Landscape Ecology*, *12*(2), 77-93. - преузето у септембру 2022

1. <https://reactjs.org/> - преузето у септембру 2022.

1. <https://reactjs.org/docs/introducing-jsx.html> - преузето у септембру 2022,

1. <https://restfulapi.net/> - преузето у септембру 2022.

1. <https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol> - преузето у септембру 2022.

1. <https://spring.io/projects/spring-boot> - преузето у септембру 2022.

1. <https://spring.io/> - преузето у септембру 2022.

1. <https://www.javadevjournal.com/spring/spring-interview-questions/> - преузето у септембру 2022.

1. <https://www.drools.org/> - преузето у септембру 2022.

1. <https://docs.spring.io/spring-data/jpa/docs/1.3.0.RELEASE/reference/html/jpa.repositories.html> - преузето у септембру 2022

1. <https://medium.com/edureka/what-is-dependency-injection-5006b53af782> - преузето у септембру 2022.

1. <https://developer.mozilla.org/en-US/docs/Web/HTTP/Status> - преузето у септембру 2022.

1. <https://reactjs.org/docs/hooks-custom.html> - преузето у септембру 2022.

1. <https://www.freecodecamp.org/news/how-to-use-axios-with-react/> - преузето у септембру 2022.

# Биографија студента

Исидора Савић рођена је у Новом Саду, 2. Септембра 1999. Ту је завршила Основну школу “Светозар Марковић Тоза”, после које уписује Гимназију “Исидора Секулић”, природно-математички смер. Након завршене средње школе, 2018. године, уписује смер Софтверско инжењерство и информационе технологије, на Факултету техничких наука. У току последњег семестра студија, у пролеће 2022. завршава стручну праксу у компанији Vega IT, после које се и запошљава тамо, на позицији *Software Engeneer*-а. Студије завршава у року, у Септембру 2022. године.

# КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

|  |  |
| --- | --- |
| Редни број, **РБР**: |  |
| Идентификациони број, **ИБР**: |  |
| Тип документације, **ТД**: | Монографска публикација |
| Тип записа, **ТЗ**: | Текстуални штампани документ |
| Врста рада, **ВР**: | Дипломски рад |
| Аутор, **АУ**: | Исидора Савић |
| Ментор, **МН**: | проф. др Синиша Николић, ФТН Нови Сад |
| Наслов рада, **НР**: | Систем за интродукцију и реинтродукцију текуница |
| Језик публикације, **ЈП**: | Српски |
| Jezik извода, **JИ**: | Српски / енглески |
| Земља публиковања, **ЗП**: | Србија |
| Уже географско подручје, **УГП**: | Војводина |
| Година, **ГO**: | 2022 |
| Издавач, **ИЗ**: | Ауторски репринт |
| Место и адреса, **MA**: | Нови Сад, Факултет техничких наука, Трг Доситеја Обрадовића 6 |
| Физички опис рада, **ФO**: | бр. поглавља 7/ страница 40/ цитата ../ табела 0/ слика 30 |
| Научна област, **НO**: | Софтверско инжењерство и информационе технологије |
| Научна дисциплина, **НД**: | Системи базирани на знању |
| Предметна одредница /  кључне речи, **ПO**: | систем базиран на правила, процена погодности станишта, представа доменског знања |
| **УДК** |  |
| Чува се, **ЧУ**: | Библиотека Факултета техничких наука, Трг Доситеја Обрадовића 6, Нови Сад |
| Важна напомена, **ВН**: |  |
| Извод, **ИЗ**: | У раду је описана веб апликација за интродукцију и реинтродукцију текуница. Описане су све технологије које су коришћене. Наведена је спецификација система, као И опис саме имплементације. На крају је приказан рад система и свих његових функционалности. |
| Датум прихватања теме, **ДП**: |  |
| Датум одбране, **ДO**: |  |
| Чланови комисије, **KO**: |  |
| председник |  |
| Члан |  |
| Ментор | Др Синиша Николић, професор, ФТН Нови Сад |
| Потпис ментора | |

# KEY WORD DOCUMENTATION

|  |  |
| --- | --- |
| Accession number, **ANO**: |  |
| Identification number, **INO**: |  |
| Document type, **DT**: | Monographic publication |
| Type of record, **TR**: | Textual material |
| Contents code, **CC**: | BSc thesis |
| Author, **AU**: | Isidora Savić |
| Mentor, **MN**: | Siniša Nikolić, PhD |
| Title, **TI**: | System for introduction and reintroduction of ground squirrels. |
| Language of text, **LT**: | Serbian |
| Language of abstract, **LA**: | Serbian / english |
| Country of publication, **CP**: | Serbia |
| Locality of publication, **LP**: | Vojvodina |
| Publication year, **PY**: | 2022 |
| Publisher, **PB**: | Author’s reprint |
| Publication place, **PP**: | Novi Sad, Faculty of Technical Sciences, Trg Dositeja Obradovića 6 |
| Physical description, **SF**: | no. of chapters 7/ pages 40/ quotes …/ tables 0/ pictures 30 |
| Scientific field, **SF**: | Software engineering and information technologies |
| Scientific discipline, **SD**: | Knowledge based systems |
| Subject / Keywords, **S/KW**: | Rule based systems, assessment of habitat suitability, representation of domain knowledge |
| **UDC** |  |
| Holding data, **HD**: | Library of the Faculty of Technical Sciences, Trg Dositeja Obradovića 6, Novi Sad |
| Note, **N**: |  |
| Abstract, **AB**: | This paper describes the web application for introduction and reintroduction of ground squirrels. Specification and implementation of the program are given in detail. The use of application is also shown, as well as all of the functionalities. |
| Accepted by sci. Board on, **ASB**: |  |
| Defended on, **DE**:: |  |
| Defense board, **DB** |  |
| president |  |
| member |  |
| mentor | dr Siniša Nikolić, professor, FTN Novi Sad |
| Mentor's signature | |