Академија струковних студија Шумадија,

одсек Аранћеловац



**Есеј:**

**ReactJS**

**Студијски програм:** Информационе технологије

**Предмет:** Стручна пракса 1

# 

**Професор:**  **Студент:**

Нела Стојиљковић Вељко Радисављевић ИТ-26/2021

Аранђеловац, децембар 2021.

# **Садржај**

[Садржај](#_xziyqxp4cmx8) 2

[Увод](#_44g2g42wrg4g) 3

[Фронт-енд Веб Девелопмент](#_cb0tm68evolb) 4

[ХТМЛ](#_m19eir7hop30) 5

[ЦСС](#_mjvsb9zbnamg) 6

[ЈаваСкрипт](#_4wd5lsxsq2s8) 8

[Ријект](#_n97ghhapfctz) 9

[Ријект ЏејЕсЕкс](#_4kxtwaknz24u) 10

[Ријект Компоненте](#_g5veh7elpa84) 11

[Ријект Хукови](#_4wt57c36w1zf) 12

[Ријект Пропови](#_l920jw35g5ws) 13

[Ријект ДОМ](#_fc63il74p626) 15

[Закључак](#_gjmd4gevcxeh) 16

[Литература](#_jtod72rlylxo) 17

# **Увод**

Живимо у времену где сваки члан породице има паметни телефон, компјутер или лаптоп. Где када интернет престане да шаље сигнал сви се дигну на ноге и буне. Исто се дешава када нешто или неки уређај кочи, а тек када апликација коју користимо да одморимо мозак, да се чујемо са неким почне да багује (*Bug*). Одма су они програмери сто су направили апликацију криви зашто кочи. React је баш тај стуб на који се апликације ослањају када се прави апликација или нека друштвена мрежа. Због своје моћи представља најпопуларнији фрејмворк (*Framework*), како због могућности тако и због просте и лако научиве синтаксе (*Syntax*) коју користи.

# 

# **Фронт-енд Веб Девелопмент**

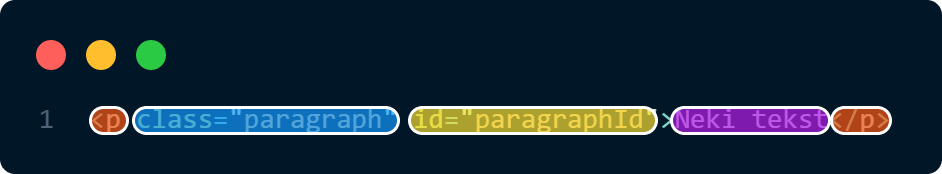
Фронт-енд веб девелопмент (*Front-end web development*) представља једну од две ставке веб девелопмента. Фронт-енд девелопмент се бави структуром, изгледом и функционалношћу веб странице. Док са друге стране имамо Бек-eнд *(Back-end*) девелопмент који се бави повезивањем, базама података и сличним стваримо што ми не можемо видети, што се практично дешава у позадини. Фронт-енд за неке лакси део, јесте оно сто можемо видети када се повежемо на веб страницу. Три главне ставке које сачињавају једну страницу су HTML (Слика 1), CSS (Слика 2) и ЈаваСкрипт (*JavaScript*) (Слика 3). HTML представља структуру странице, CSS побољшава изглед наше странице и за крај имамо JavuScript која ће употпунити наш сајт са функционалношћу.

(Слика 1 - HTML лого) (Слика 2 - CSS лого) (JavaScript лого)

# **ХТМЛ**

Хајпертекст Маркап Језик (*Hypertext Markup Language*) или само хтмл представља срж самог веб сајта. Нешто основно без чега се не може, нешто као кости у човеку. Израда је почела 1989. године од стране Тим Бернерс Лија (*Tim Berners Lee*). Особе која је направила први претраживач (*Browser*) World Wide Web познатији као www. Угледом на Стандардни Генерализовани Маркап Језик (*Standard Generalized Markup Language*) или само сгмл (*STML*), где сваки елемент има отворено и затворено обележје (*Tag*) који репрезентује одређено значење веб страници. По речима Џули Ц. Мелони (*Julie C. Meloni*) хтмл је језик за описивање како текста, графике и осталих садржаја, информације су организоване и спојене заједно. Na sledeћем примеру (Слика 4) види се како изгледа један параграф унутар хтмл кода.

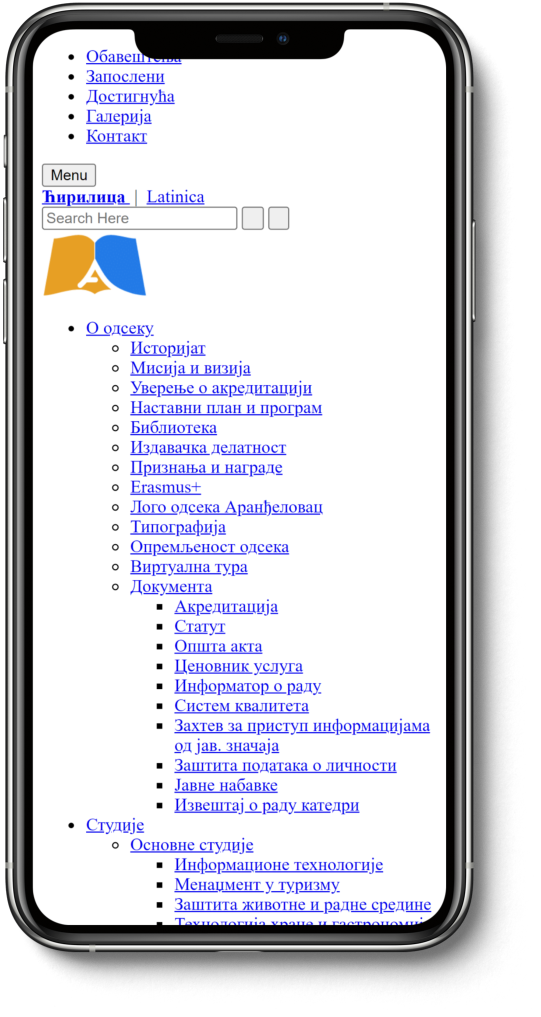


(Слика 4 - Параграф у хтмл коду)

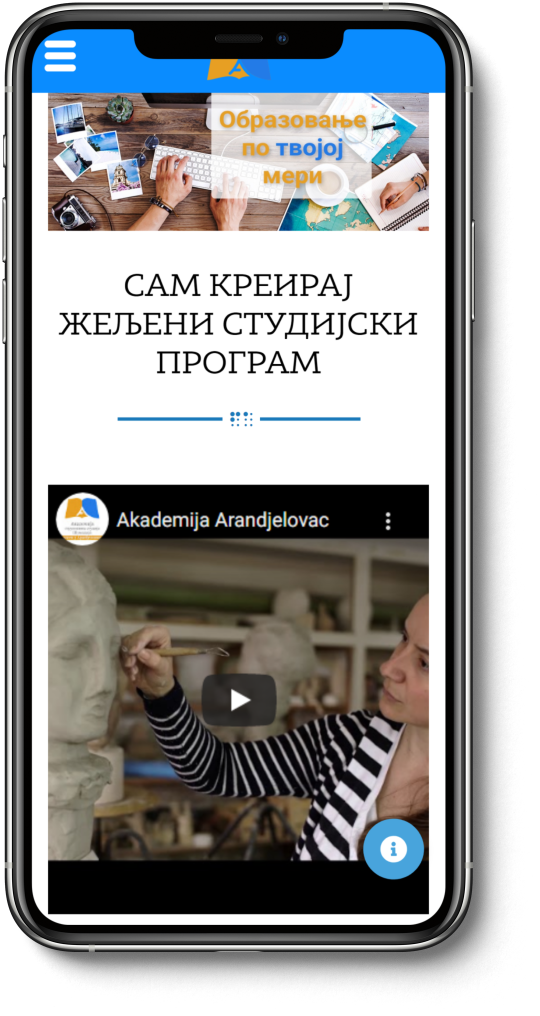
Код унутар квадрата обојеном наранџастом бојом представља обележје (*Tag*) самог параграфа. Сваки елемент има јединствено обележје, тако имамо <p></p>, <h1></h1>, <div></div> и многе друге. Код унутар квадрата обојеном плавом бојом представља класу (*Class*) или јединствен позив који нам унутар CSS кода омогућава да тачно тај параграф променимо како желимо, без да променимо остале параграфе. Код унутар жутог квадрата представља јединствени ајди (*Id*) са којим можемо приступити унутар JavaScript кода, којег можемо манипулисати по жељи. И за крај имамо код унутар љубичастог квадрата који нам представља текст који ће бити приказан на веб страници.

# **ЦСС**

СиЕсЕс (CSS) или Каскадинг Стајл Шит (*Cascading Style Sheet*) представља могућност да нашу веб страницу из досадног, црно белог хтмл кода претворимо у нешто много лепше и много функционалније и лакше за коришћење. Помоћу одређивања класа (*Class*) и обележја (*Tag*) које смо претходно декларисали унутар хтмл (html) документа, можемо управљати елементима како желимо. Повећавати им величину, ширину, боју и многе друге ствари. (Слика 5) Репрезентује сајт без CSS-а.



(Слика 5 - Веб сајт без CSS-a)

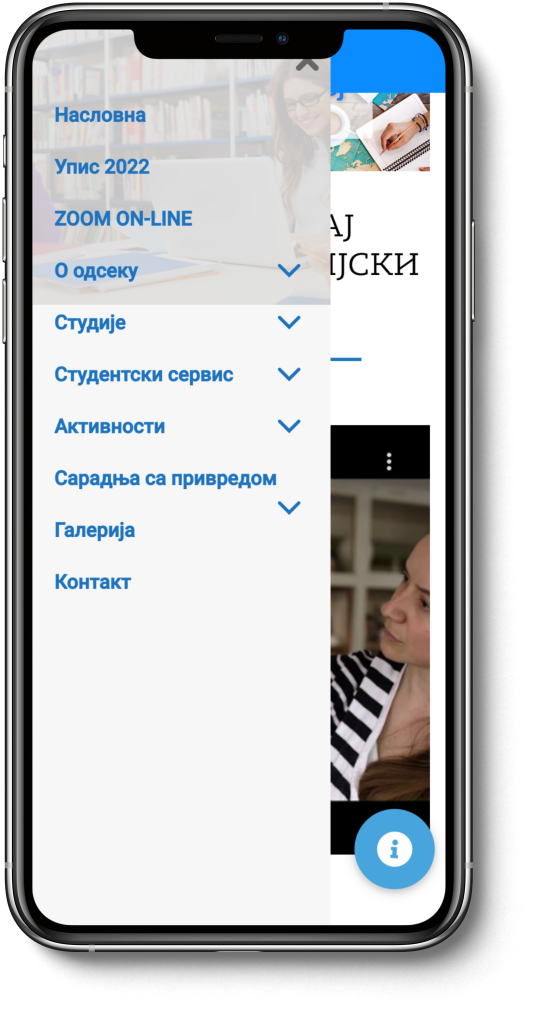


(Слика 6 - Веб сајт са CSS)

Док (Слика 6) приказује исти веб сајт само са CSS. Као што се види CSS може направити тоталну разлику када се прича о лепоти једног веб сајта.

# **ЈаваСкрипт**

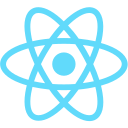
ЈаваСкрипт (*JavaScript*) по речима Дејвида Фланангана (*David Flanangan*) JavaScript је програмски језик веба. Огромна већина данашњих модерних веб страница користи JavaScript и сви модерни веб претраживачи (*Browsers*) на радним површинама (*Desktop*), као и игрице, конзоле таблети и паметни телефони користе JavaScript, што JavaScript прави најприступачнијим програмским језиком у историји. Не толико тежак за почетнике, а отвара врата за многоброје позиције. На веб страници JavaScript прави разлику између лепе веб странице од оне функционалне веб странице. JavaScript уз помоћ Media Queries нам омогућава да навигацију (*Navigation*) направимо приступачном на малим екранима, тачније паметним телефонима који због своје мале ширине не могу приказати исту навигацију (*Navigation*) као и рачунари на широким екранима.



(Слика 7 - Sidebar Menu)

# **Ријект**

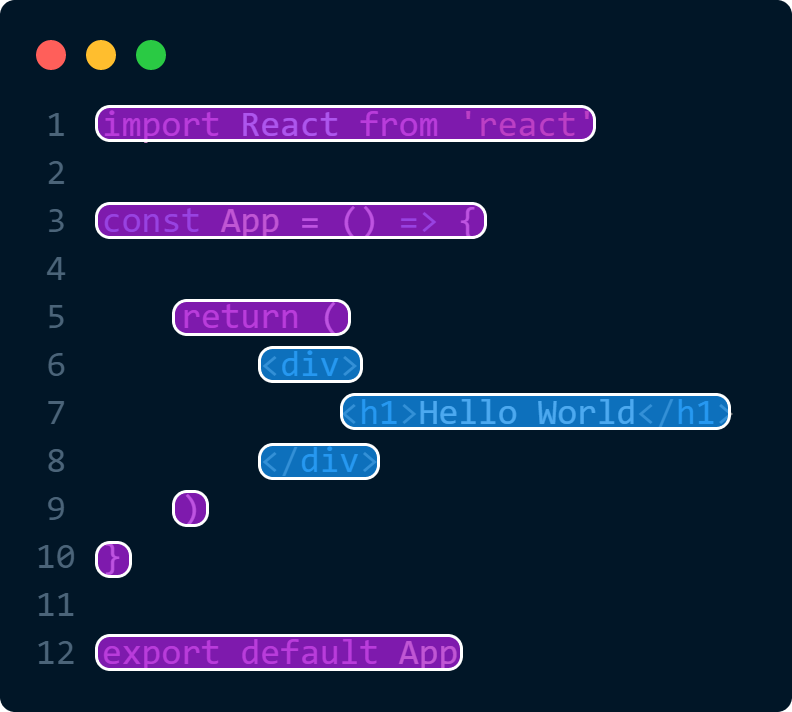
РијектЏејЕс (*ReactJS*) или само Ријект (*React*) направљен 2013. године од стране Фејсбука (*Facebook*), док сама идеја потиче од запосленог Џордана Волкеа (*Jordan Valke*) радника, представља JavaScript фрејмворк (*Framework*), за неке и библиотеку (*Library*) за изградњу корисничких интерфејса (*User Interfaces*). Циљ и сам задатак фрејмворка (*Framework*) представља лако манипулисање компонентама (*Components*), лако креирање истих, као и убрзавање тачније смањење успорености *(Lag*) саме веб апликације. Неки од многобројних разлога зашто се веб девелопери се одлуце баш за React су: лак је за учење, лак је за разумевање, лако манипулисање компонентама, као и лако креирање истих, брзо учитавање (*Render*) истих компоненти итд…



(Слика 8 - React лого)

# **Ријект ЏејЕсЕкс**

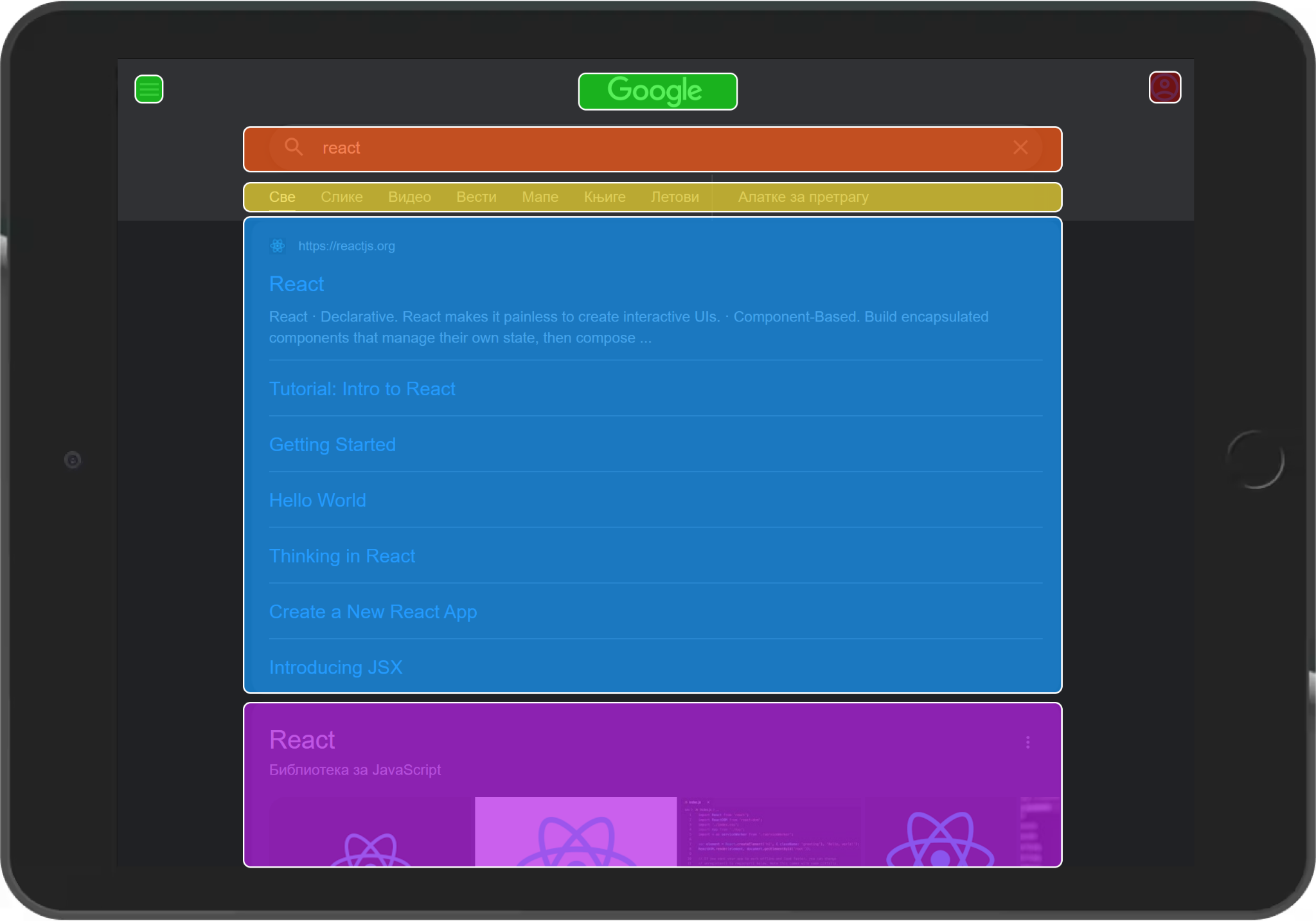
Ријект ЏејЕсЕкс (*React JSX*) представљају синтаксу (*Syntax*) (комбинацију *HTML* и *JavaScript* синтакси), направљен од Фејсбука (*Facebook*), ради лакшег коришћења и разумевања самог кода. Корене вуче од HTML кода где свако обележје (*Tag*) почиње и завршава се. Садржи исте елементе као и сам HTML, као и могућност уметања (*Nesting*) самог JavaScript дела, тачније функција унунтар једног елемента. Као на примеру у наставку (Слика 9), где код унутар квадрата обојеним плавом бојом представљају HTML део кода, док код унутар квадрата обојеним љубичастом бојом представља JavaScript део.



(Слика 9 - JSX код)

# **Ријект Компоненте**

Ријект Компоненте (*React Components*) представљају независно парче кода, које се може касније користити много пута. Саму моћ компоненти (*Components*) представљају React Пропови (*Props*), који саму компоненту могу променити независно од употребе, где свака компонента има исту срж, тачније исту примену само различит прикзан податак. На следећем примеру (Слика 10) moжемо видети како то све изгледа на једном уређају, у овом слуцају iPad Pro. Сваки квадрат обојен другачијом бојом представља независну компоненту (*Component*), која унутар ње садржи остале елементе које попуњавају исту.



(Слика 10 - Компоненте (Components) на iPad Pro)

# **Ријект Хукови**

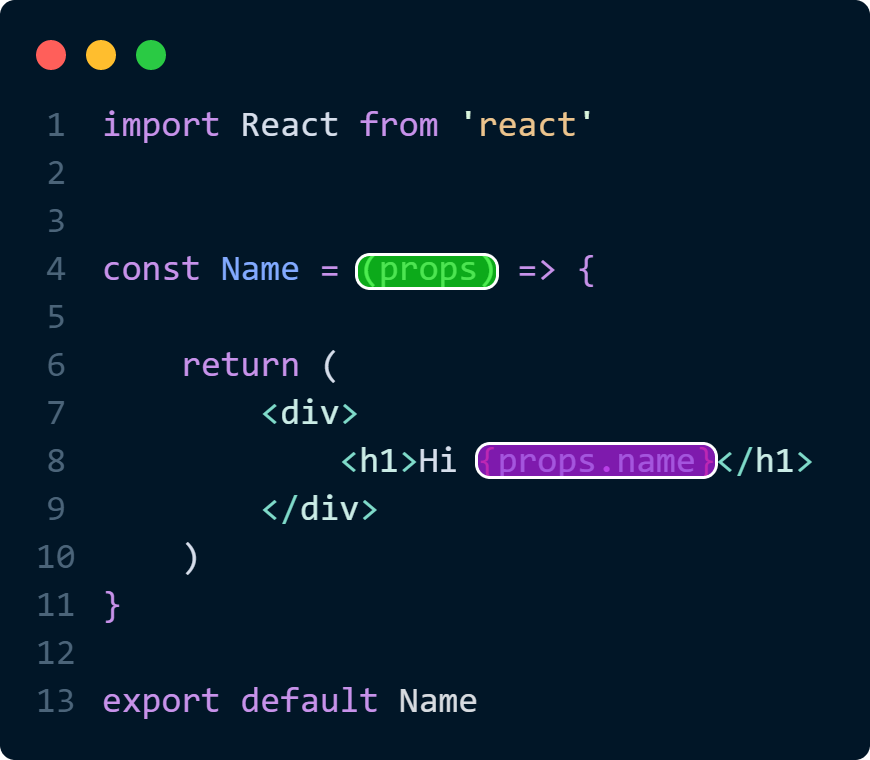
React Хукови (*Hooks*) просто речено дају до знања React-у шта се догађа на веб апликацији и који податак треба сачувати и променити тачније ажурирати (*Update*) на веб страници. Постоје много хукова (*Hooks*), један од најпопуларнијих и најкориснијих јесте јузстејт хук (*useState Hook*), који прати стање и ажурира по потреби. У следећем примеру ћемо видети како изгледа код за дугме (*Button*) које након једног клика (*Click*) повећава испис броја за један. (Слика 11) приказује код компоненте где квадрат обојен плавом бојом нам приказује код који нам омогућава да користимо јузстејт хук (useState Hook) унутар ове компоненте. Квадрат обојен наранџастом бојом представља код који нам приказује тренутну стање, у овом случају почетно (*Default*) прадставља нулу. Квадрат обојен љубичастом бојом представља упит React-у, који би значио Хеј React, ја желим помоћу ове компоненте да ажурирам (*Update*) тренутно стање, да ли ми дозвољаваш то? И за сам крај имамо квадрат обојен жутом бојом који нам представља код који на клик (*Click*) дугмета (*Button*) позива функцију која узима тренутно стање и на то додаје број.



(Слика 11 - Kod за повећавања броја за 1)

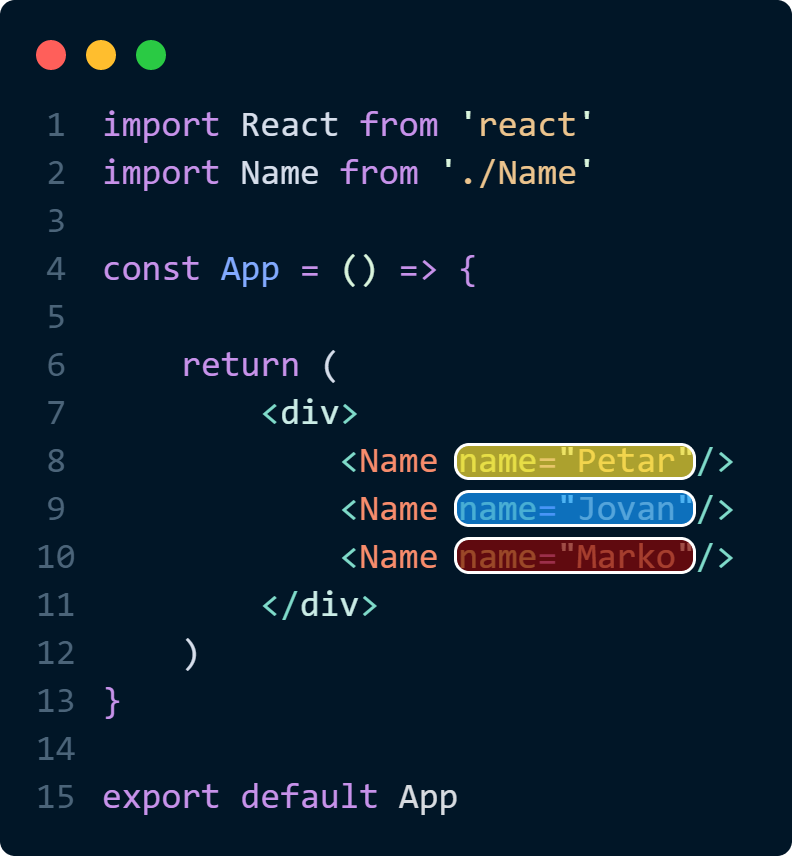
# **Ријект Пропови**

React Пропови (*Props*) представљају променљиво (*Changeable*) и поновно коришћене (*Reusable*) делове које можемо користити у зависности од ситуацијуа, као и од задатака које имамо. Већином Пропови (*Props*) имају примену у виду пребацивања неких података (*Data*), као у једном од примера јесте листа задатака. По примеру Антони Ацомацоа *(Anthony Accomazzo*), Ари Лернера (*Ari Lerner*), Нејт Муреја (*Nate Murray*), Клеј Алсопа (*Clay Allsopp*), Дејвид Гутмана (*David Guttman*) и Тајлер ЕмГиниса (*Tyler McGinnis*). Ми желимо да мoдификујемо (*Modify*) нашу Продукт компоненту (*Product Component*) тако да више не користи статичне (*Static*), само кодиране атрибуте (*Attribute*). Већ желимо да буде у могућности да прихвати податак свог родитења (*Parent*) Продукт листу (*Product list*). Подешавањем структуре наше компоненте овим начином омогућава нам да свака Продукт Компонента (*Product Component*) има свој уникантан продукт (Product). У следећем примеру можемо видети употребу пропова (*Props*) у много лакшем и једноставнијем коду (Слика 12). Где имамо квадрат обојен зеленом бојом који представља поље за унос аргумента (*Arguments*). Док квадрат обојен љубичастом бојом представља позив самих пропова као и њихову “промену”.



(Слика 12 - Name компонента (Component))

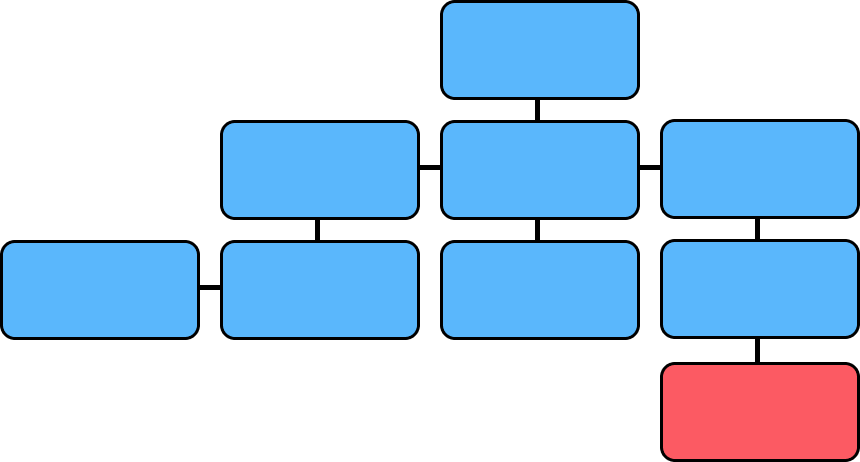
У наредној слици (Слика 13) видимо компоненту (*Components*) Ep (*App*), која садржи наше компоненте, у овом примеру три компоненте под називом Нејм (*Name*), које смо претходно направили и у којима смо позвали саме пропове (*Props*). Квадрати обојеноми жутом, плавом и црвеном бојом представљају ту промену пропова (*Props*) у зависност од потребе. Као што је овде приказано у свакој компоненти имамо различито име позвано атрибутом (*Attribute*) нејм (*Name*).



(Слика 13 - App компонента (Component))

# **Ријект ДОМ**

По речима Робина Вируха (*Robin Wieruch*) унутар главног фајла (*File*), који садржи све наше компоненте (*Components*) и који исте приказује (Render), ту је и још једна уметнута (*Imported*) библиотека (*Library*) названа react-dom, у којој ReactDOM.render() фунцкија користи HTML срж да је замени са JSX. Процес интегрира React унутар HTML. ReactDOM.render() захтева два аргумента (*Arguments*), Први је да прикаже ЈSX. To pravi molbu (*Instance*) наше Аpp компоненте (*Components*). То би значило да ће React ажурирати само тамо где је неопходно. Рецимо да имамо неку објаву на фејсбуку (*Facebook*) и желимо да оставимо коментар. Након остављеног коментара наша страница се поново учита (*Reload*). То се десило да React не прави виртуални дом (*Virtual DOM*), који прати и реагује на промену и извршава тамо где је неопходно. На следећој слици (Слика 14) видимо приказ структуре компоненти (*Components*), њихове повезаности и реакцију ReactDOM-а на промену. Као што се види на слици све компоненте су биле плаве, након промене боје једне од компоненти, само ће се она променити и ажурирати (*Update*), док ће остале остати нетакнуте.



(Слика 14 - Структура компоненти (Components))

# **Закључак**

React са разлогом води на лествици најпопуларнијих и најомиљенијих фрејмворкова (*Frameworks*) код девелопера. Толико лакши за учење, а тако добар посао се може урадити са њим. Свако зна какав је и колико снажан је фејсбук (*Facebook*) и баш је он направњен са React. Мислим да то довољно говори о безбедности тачније, какав је ослонац када је Фејсбук (*Facebook*) направљен са њим.

# **Литература**

1. Accomazzo, Anothony, et al. *Fullstack React: The Complete Guide to ReactJS and Friends*. San Francisco, Fullstack.io, 2017.
2. Banks, Alex, and Eva Porcello. *Learning React: Functional Web Development with React and Redux*. Sebastopol, O’Reilly Media, 2017.
3. Bugl, Daniel. *Learn React Hooks*. Birmingham, Packt Publishing, 2019.
4. Copes, Flavio. *The React Beginner's handbook*.
5. Flangan, David. *Javascript: The Definitive Guide*. 6th ed., O'reilly.
6. Mardan, Azat. *React Quickly: Painless Web Apps with React, JSX, Redux, and GraphQ*. Manning.
7. Meloni, Julie C. *Sams Teach Yourself HTML, CSS, and JavaScript All in One*. Indianapolis, Pearson Education, 2012.
8. Wieruch, Robin. *The Road to Learn React: Your Journey to Master Plain Yet Pragmatic React. Js*. Leanpub, 2020.