Зборник радова Факултета техничких наука, Нови Сад

UDK: 004.9
DOI: https://doi.org/10.24867/08BE18Tacic

NATIVESCRIPT ОКВИР ЗА РАЗВОЈ МОБИЛНИХ АПЛИКАЦИЈА

NATIVESCRIPT MOBILE APP DEVELOPMENT FRAMEWORK

Дарко Тачић, *Факултет техничких наука, Нови Сад*

Област – ЕЛЕКТРОТЕХНИЧКО И РАЧУНАРСКО ИНЖЕЊЕРСТВО

Кратак садржај – *У овом раду анализиран је NativeScript радни оквир за развој мобилних апликација. Затим су специфицирани захтеви и дизајн мобилне апликације за евиденцију пацијената. На крају је специфицирана апликација имплементирана и тестираана у NativeScript радном оквиру.*

Кључне речи – *NativeScript, Cordova, Sidekick, Playground, JWT, MyPatientCare, MyDoctorCare*

Abstract – *This paper analyses the NativeScript mobile app development framework. Then it specifies requirements and design of a mobile app for managing patients. Finally, it describes the implementation and testing of the specified app.*

Keywords – *NativeScript, Cordova, Sidekick, Playground, JWT, MyPatientCare, MyDoctorCare*

1. УВОД

Мобилне технологије спадају међу технологије које се најбрже развијају. Тренутно 66.53% светског становништва поседује мобилни телефон [1]. У циљу олакшавања програмерима креирања мобилних апликација развијена је NativeScript технологија. У питању је платформа која се користи за писање нативних вишеплатформских мобилних апликација уз помоћ познатих веб технологија (HTML, CSS и JavaScript).

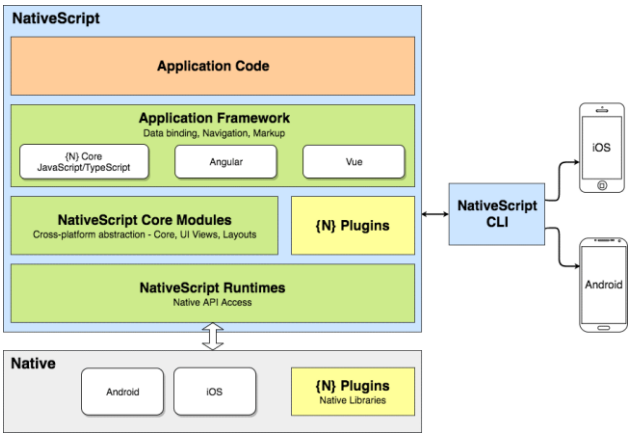
Поменута технологија ће бити представљена кроз две мобилне апликације, чија је главна улога да воде евиденцију прегледа пацијената у болници.

2. МОБИЛНЕ ПЛАТФОРМЕ

Мобилне апликације се могу груписати у четири велике категорије по врсти и окружењу: Native – Android и iOS, Hybrid – PhoneGap и Cordova, cross-compiled – Xamarin, JIT – NativeScript. Најзначајнија карактеристика ненативних апликација јесте могућност писања кода који може да се извршава на више платформи само једном Технологије које ово омогућавају називамо *cross-platform.*

2.1 NativeScript

NativeScript [2], као један од савремених представника *cross-platform* технологија, спада међу JIT компајлиране апликације и користи JavaScript виртуелну машину као окружење за рад. NativeScript се састоји од следећих компоненти (слика 1): извршно окружење (енг. runtimes), модули (енг. core modules), кориснички интерфејс (енг. command line interface) и додаци (енг. plugins).



Слика 1. *Компоненте које чине NativeScript [3]*

2.2. JSON Web Token

JWT (JSON Web Token) представља стандард за креирање веб токена ради ауторизације корисника. Након што се клијент успешно аутентификује, сервер генерише овај токен и прослеђује га клијенту.
JWT се састоји из три основна дела: заглавље (енг. header), садржај (енг. payload) и потпис (енг. signature).

3. СПЕЦИФИКАЦИЈА ЗАХТЕВА

Захтеви које апликација треба да испуни се могу груписати у функционалне и нефункционалне.

3.1 Функционални захтеви

У раду су приказани функционални захтеви као случајеви коришћења. Корисници система (лекари и пацијенти) имају низ заједничких и појединачних случајева коришћења
Заједнички случајеви коришћења су:

- Логовање
- Регистрација
- Приказ посета
- Приказ детаља посете

Појединачни случајеви коришћења за пацијента су:

- Креирање посете
- Бирање третмана
- Бирање лекара

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био др Стеван Гостојић.

- Бирање датума
- Бирање времена
- Приказ контакта
- Слање email-а
- Слање sms поруке
- Телефонски позив
- Приказ информација здравствене установе

Појединачни случајеви коришћења за лекара су:

- Приказ посета
- Приказ детаља посета
- Потврда посете
- Одбијање посете
- Приказ годишњих одмора
- Додавање годишњег одмора
- Ажурирање годишњег одмора
- Приказ радних времена
- Додавање радног времена
- Ажурирање радног времена
- Ажурирање профила

3.2 Нефункционални захтеви

Најбитнији нефункционални захтеви које апликација треба да испуни су:

- Перформансе
- Безбедност
- Поузданост

4. СПЕФИЦИКАЦИЈА ДИЗАЈНА

Спецификацију дизајна апликације чине статички и динамички модели.

4.1 Статички модели система

Дијаграм класа представља врсту UML дијаграма који описује структуру система приказивањем класа система, њихових атрибута, операција (метода) и односа између објеката. Дијаграм класа за апликацију за вођења евиденције пацијената приказан је на слици 2.

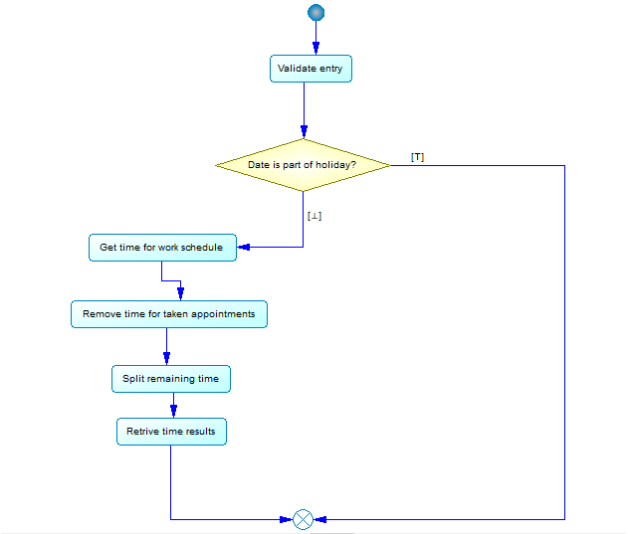
4.2 Динамички модели система

Дијаграм активности је дијаграм који истиче ток контроле од активности до активности. Користи се за детаљно описивање посебних случајева коришћења.

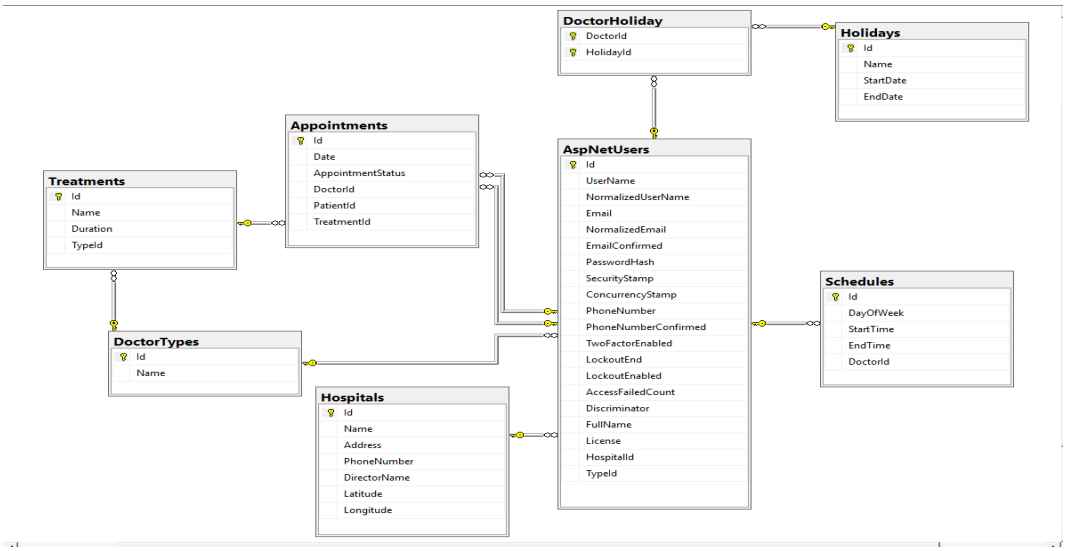
Процес добављања слободних термина започиње валидацијом улазних података.

Уколико је валидација успешна, поставља се питање да ли за жељени датум има слободних термина. Следи провера да ли жељени датум припада неком годишњем одмору. Уколико не припада, добављају се сви термини за тај дан.

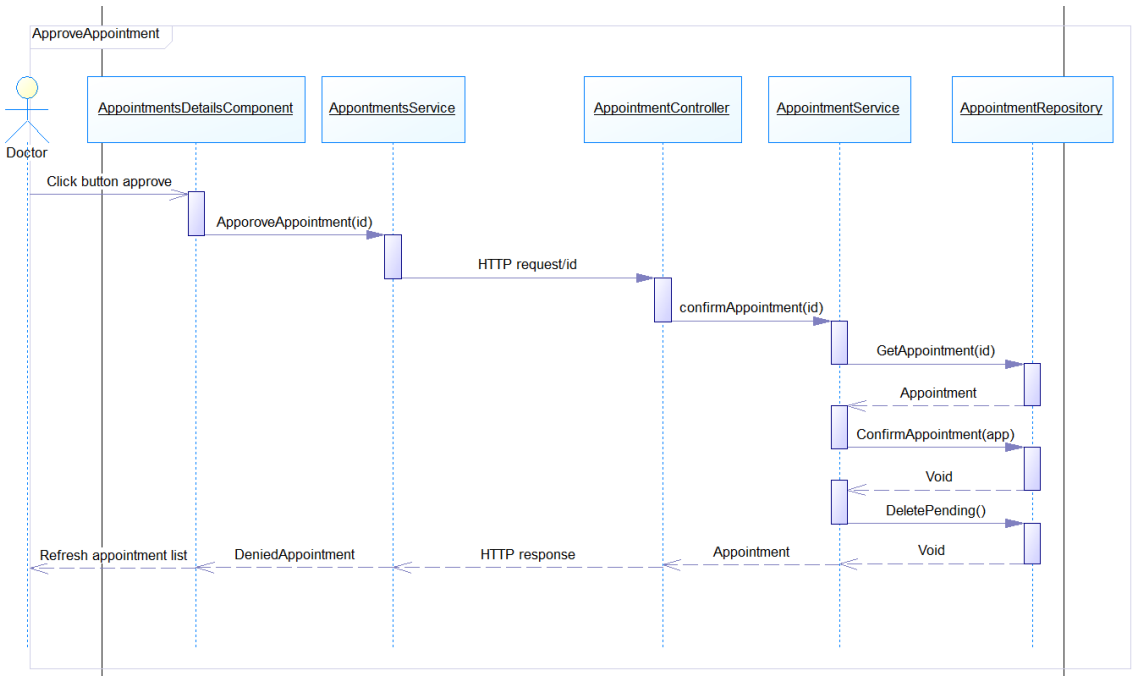
Међутим, уколико у бази већ постоје заказани термини за тај датум, потребно је елиминисати заузете термине из листе слободних времена. Коначно, кориснику се достављају одговарајући подаци (слика 3).



Слика 3. *Дијаграм активности за добављање листе слободних термина*



Слика 2. *Дијаграм класа за пројекат вођење евиденције пацијената*



Слика 4. *Дијаграм секвенце за процес потврде посете лекару*

Дијаграм секвенце представља динамичку слику односа између објеката. Ова динамика се види из порука које они размењују. У пројекту је коришћен дијаграм секвенце за додатни опис случаја коришћења потврда посете лекару.

Процес потврде посете започиње притиском лекара на дугме "Approve Reservation". Даље, путем AppointmentsService-a се шаље HTTP захтев ка Backend серверу. Endpoint за овај захтев се обрађује у класи AppointmentController. Уколико је валидација података успешна, позива се confirmAppointment метода која се налази у класи AppointmentService. Уколико тражени Appointment постоји у систему, врши се промена стања у Approved, као и брисање свих оних резервација које су временски пресецале претходно потврђену резервацију.

Клијент на крају добија повратну информацију на освеженом екрану (Слика 4).

4.3 Опис сервиса система

Важно је дати осврт на сервис класе које сачињавају пословну логику апликације за вођење евиденције пацијената. Сервис класе су: UserService, TreatmentService, ScheduleService, HospitalService, ContactInfoService, HolidayService и AppointmentService.

5. ИМПЛЕМЕНТАЦИЈА

За реализацију пројекта коришћено је низ софтверских алата, који су међусобно повезани. Софтверски алати примењених у току израде пројекта су:

- Visual Studio [4]
- Visual Studio Code [5]
- NativeScript Playground [6]
- NativeScript Sidekick [7]
- SQL Management Studio [8]
- Android Virutal Device Manager [9]

Имплементација JSON Web токена обухвата више фаза. Пре него што сервер генерише токен, потребно је да у конфигурацији сервера омогућимо овај вид комуникације. Жељени циљ се постиже додавањем CORS (Cross-Origin Resource Sharing) који представља механизам комуникације путем HTTP протокола између сервера различитог порекла. Порекло сервера може да зависи од његовог домена, протокола комуникација или од вредности порта.

```
        // This method gets called by the runtime.
Use this method to configure the HTTP request
pipeline.
        /// <summary>
        /// The Configure
        /// </summary>
        /// <param name="app">The
app<seecref="IApplicationBuilder"/></param>
        /// <param name="env">The
env<seecref="IHostingEnvironment"/></param>
        public void Configure(IApplicationBuilder
app, IHostingEnvironment env)
        {
            if (env.IsDevelopment())
            {
                app.UseDeveloperExceptionPage();
            }
            app.UseAuthentication();
            app.UseCors(builder =>
            builder.AllowAnyOrigin()
            .AllowAnyHeader()
            .AllowAnyMethod());
            app.UseMvcWithDefaultRoute();
            }
```

Листинг 1. *Конфигурација сервера за CORS*

6. ДЕМОНСТРАЦИЈА

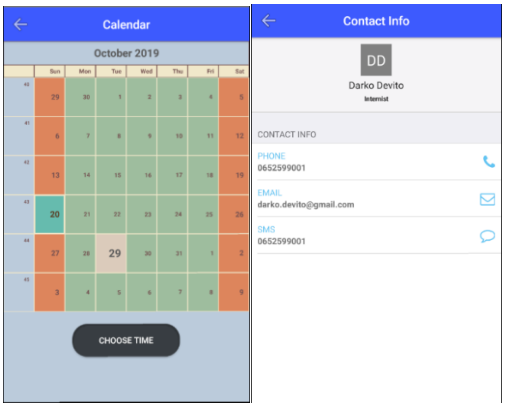
Апликација је демонстрирана приказом графичког корисничког интерфејса.

6.1 MyPatientCare

MyPatientCare је апликација за пацијенте. Основне функционалности које пацијент извршава помоћу ове апликације су заказивање прегледа, контактирање лекара и преглед информација о здравственој установи.

Основни екрани које чине MyPatientCare апликацију:

- Login Component
- Register Component
- Appointments Component
- Appointments Details Component
- Treatments Component
- Doctors Component
- Calendar Component (слика 5 – лево)
- Contact Component
- ContactInfo Component (слика 5 – десно)
- AboutUs Component



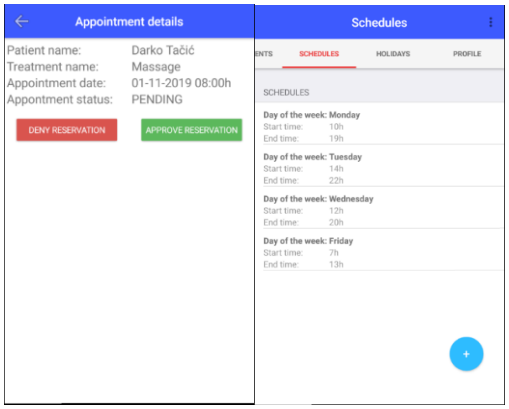
Слика 5. *Изглед екрана са календаром и екрана са контакт подацима*

6.1 MyDoctorCare

MyDoctorCare је апликацију за лекаре. Основне функционалности које лекар извршава су потврда или одбијање прегледа, ажурирање радног времена и ажурирање одсуства.

Основни екрани које чине MyDoctorCare апликацију су:

- Appointment Details Component (слика 6 – лево)
- Schedule Component (слика 6 – десно)
- Holiday Component
- Profile Component



Слика 6. *Изглед екрана за потврду или одбијање резервације и екрана за приказ радног времена*

7. ЗАКЉУЧАК

У раду је приказан NativeScript оквир, као алтернативни начин за развој мобилних апликација.

NativeScript сам по себи не нуди много нових могућности у поређењу са сличним технологијама. Међутим, захваљујући алатима који га окружују, комплетиране су све фазе развоја мобилних апликација.

Упркос високој интуитивности NativeScript-a, недостатак документације умногоме отежава његову примену. Ова мана се може објаснити његовој малој популарности, с обзиром да је оквир релативно нов. Ипак, тим који одржава NativeScript не одустаје од свог производа и наставља да надограђује постојеће функционалности.

8. ЛИТЕРАТУРА

[1] https://bankmycell.com/blog/how-many-phones-are-in-the-world (*Приступљено:17.09.2019.)*

[2] *NativeScript (2019), Progres, Приступљено:17.10.2019.* [Online] Доступно: https://docs.nativescript.org/angular/core-concepts/technical-overview

[3] M. Branstein, N. Branstein, *"The NativeScript ebook"*, 2016

[4] *VisualStudio (2019), Microsoft, Приступљено:05.08.2019.* [Online] Доступно: https://visualstudio.microsoft.com/

[5] *VisualStudioCode (2019), Microsoft, Приступљено:03.09.2019.* [Online] Доступно: https://code.visualstudio.com/

[6] *NativeScript Playground (2019), Progres, Приступљено*: *30.09.2019.* [Online], Доступно: https://play.nativescript.org/

[7] *NativeScript SideKick (2019), Progres, Приступљено*: *28.09.2019.* [Online], Доступно: https://www.nativescript.org/nativescript-sidekick

[8] *SQL Management Studio (2018), Microsoft, Приступљено*: *11.10.2019.* [Online], Доступно: https://docs.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver15

[9] *Android virtual device manager (2019), Google, Приступљено*: *15.10.2019.* [Online], Доступно: https://developer.android.com/studio/run/managing-avds

Кратка биографија:

Дарко Тачић рођен је 15.10.1994. године у Суботици. године. Гимназију „Светозар Марковић“ у Суботици завршио је 2013. године. Исте године уписао се на Факултет техничких наука, одсек Рачунарство и аутоматика. У четвртој години бира смер Интернет и електронско пословање. Положио је све испите предвиђене планом и програмом. Мастер студије је уписао 2017. године на истом одсеку, модул Електронско пословање.