Mrežni i mobilni operacijski sustavi - Drugi seminarski rad i projekt:

Nativescript senzor aplikacija
“Skeniranje otiska prsta”

Sadržaj

Opis aplikacije

Primarna funkcija aplikacije je da je u mogućnost skenirati otiske prstiju, no to nije jedina funckionalnost koju ima.
Uz to, može se provjeravati dostupnost te koristiti fallback opcije. Izgled aplikacije je jednostavan, no ispunjava svoju svrhu.
Aplikacija je pisana u Native i TypeScriptu, uz dodatak plugina nativescript-fingerprint-auth i tns-core-modules.

Programski kod

Kod je pisan u TypeScript-u i Native-u. Za početak sam koristila gotov template prazne TypeScript aplikacije:

```
tns create SkeniranjeOtiskaPrsta --template @nativescript/template-blank-ts
```

I dodala sljedeće ekstenzije:

```
tns plugin add nativescript-fingerprint-auth
tns plugin add tns-core-modules
tns platform add android
```

App.ts

U App.tsu dodajemo template iz Core-a NativeScript paketa i povezujemo root aplikacije pomoću čega možemo naći i pokrenuti ostatak koda.

```
import * as application from "tns-core-modules/application";
application.run({ moduleName: "app-root" });
```

App.css

U app.ccsu je definiran izgled aplikacije, veličina i boja gumbi, titlova i slično.

```
page {
 background-color: #F4F4F4;
}

label {
 font-size: 14;
 }

 .tab-content {
 color: #808080;
 padding: 20;
}

.title {
 font-size: 18;
 margin: 0 0 10 0;
 color: #3c3c3c;
}

.status {
 border-radius: 8;
 font-size: 16;
 margin: 20;
 padding: 18 12;
 color: #444444;
 background-color: #cccccc;
 border-color: #aaaaaa;
 border-width: 1px;
}

button {
 background-color: #3598db;
 padding: 8 0;
 margin: 10;
 margin-bottom: 20;
 font-size: 16;
 border-radius: 4;
}

.button {
color: #ffffff; }
```

main-page.ts:

```
import * as observable from "@nativescript/core";
import * as pages from "@nativescript/core/ui";
import {HelloWorldModel} from "./main-view-model";
import { Observable } from "@nativescript/core";
export function pageLoaded(args: observable.EventData) {
    let page = <pages.Page>args.object;
    page.bindingContext = new HelloWorldModel();
}
```

U main-page.tsu su navedeni svi importi gotovih templatea i modula. Također definiramo gdje se nalazi view unutar aplikacije.
Na kraju, definiramo funkciju pageLoaded koja stvara novu stranicu i filla je view-om HelloWorld.

main-page.xml:

U Main-page.xmlu definiramo glavne elemenate aplikacije:
Prisutni su sljedeći elementi:

- TabView - Raspored elemenata aplikacije u obliku prozora (eng. “tabs”)
- TabViewItem - stavke koje će se oblikovati u tabove: Naslov i View
- ScrollView - view u kojem prst služi kao klizač i unutar kojem možemo pronaći: Upute/informacije o funkcionalnostima te gumbe koje aktiviraju te funkcionalnosti.

```
<Page xmlns="http://schemas.nativescript.org/tns.xsd" loaded="pageLoaded">
 <TabView>
  <TabView.items>
   <TabViewItem title="Demo">
    <TabViewItem.view>
     <ScrollView>
      <StackLayout class="tab-content">
       <Label text="{{ status }}" class="status" textWrap="true"
style="text-align: center"/>
       <Label text="Checking availability" class="title"/>
       <Button text="Je li dostupno?" tap="{{ doCheckAvailable }}"
class="button" />
       <Label text="Detektira promijene u bazi." textWrap="true"/>
       <Button text="Je li biometrijski ID promijenjen?" tap="{{
doCheckFingerprintsChanged }}" class="button" />
       <Label text="Skeniranje otiska prsta" class="title"/>
```

```
       <Label text="Ako skeniranje nije uspješno, koristite lozinku."
textWrap="true"/>
       <Button text="verify with passcode fallback" tap="{{ doVerifyFingerprint
}}" class="button" />
       <iOS>
        <Button text="verify with custom fallback" tap="{{
doVerifyFingerprintWithCustomFallback }}" class="button" />
       </iOS>
       <Android>
        <Label text="Ukoliko imate grešku, probajte ponovno" textWrap="true"/>
        <Button text="verify with custom UI" tap="{{
doVerifyFingerprintWithCustomUI }}" class="button" />
        </Android>
       </StackLayout>
      </ScrollView>
     </TabViewItem.view>
    </TabViewItem>
   <TabViewItem title="About">
    <TabViewItem.view>
     <StackLayout class="tab-content">
      <Image margin="10" src="~/res/telerik-logo.png" />
      <Label text="Fingerprint Auth plugin demo" class="title"/>
      <Label text="Omogućuje Vam koristiti skener otiska prsta."
textWrap="true"/>
      <Label text=" "/>
      <Label text="Zamijenjuje tradicionalne lozike." textWrap="true"/>
     </StackLayout>
    </TabViewItem.view>
   </TabViewItem>
  </TabView.items>
 </TabView>
</Page>
```

main-view-model.ts:

```
import { BiometricIDAvailableResult, FingerprintAuth } from
"nativescript-fingerprint-auth";
import { Observable } from "tns-core-modules/data/observable";
import { alert } from "tns-core-modules/ui/dialogs";

export class HelloWorldModel extends Observable {
 private fingerprintAuth: FingerprintAuth;
 public status: string = "Pritisnite gumb";
 private static CONFIGURED_PASSWORD = "MyPassword";
```

```
constructor() {
 super();
 this.fingerprintAuth = new FingerprintAuth();
}

public doCheckAvailable(): void {
 this.fingerprintAuth
 .available()
 .then((result: BiometricIDAvailableResult) => {
  console.log("doCheckAvailable result: " + JSON.stringify(result));
  this.set(
   "status",
   "Biometric ID available? - " +
    (result.any ? (result.face ? "Face" : "Touch") : "NO")
   );
  })
  .catch(err => {
   console.log("doCheckAvailable error: " + err);
   this.set("status", "Error: " + err);
  });
 }

public doCheckFingerprintsChanged(): void {
 this.fingerprintAuth
  .didFingerprintDatabaseChange()
  .then((changed: boolean) => {
   this.set( "status", "Je li biometrijski ID promijenjen? - " + (changed ?
"YES" : "NO")
   );
  });
 }

public doVerifyFingerprint(): void {
 this.fingerprintAuth
  .verifyFingerprint({
   title: "Upišite lozinku",
   message: "Skenirajte prst",
   authenticationValidityDuration: 10
  })
  .then(() => this.set("status", "Biometric ID / passcode OK"))
  .catch(err => {
   alert({
    title: "Biometric ID NOT OK / canceled",
    message: JSON.stringify(err),
    okButtonText: "Okay"
   });
  });
```

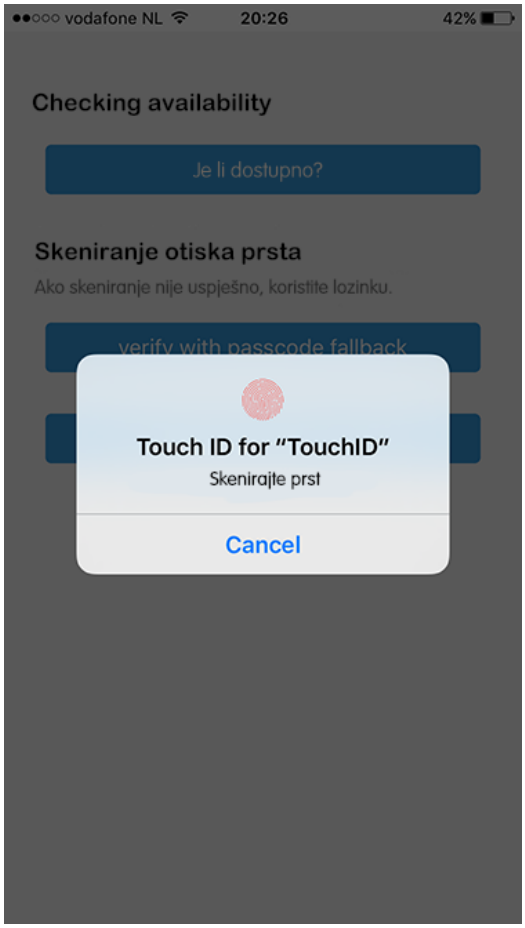
```
 }

public doVerifyFingerprintWithCustomUI(): void {
 this.fingerprintAuth
  .verifyFingerprint({
   message: "Skenirajte prst",
   useCustomAndroidUI: true
  })
  .then((enteredPassword?: string) => {
   if (enteredPassword === undefined) {
    this.set("status", "Biometric ID OK");
   } else {
    if (enteredPassword === HelloWorldModel.CONFIGURED_PASSWORD){
     this.set("status", "Biometric ID OK, using password");
    } else {
     this.set(
      "status",
      `Wrong password. Try '${HelloWorldModel.CONFIGURED_PASSWORD}' 😉`
     );
    }
   }
  })
  .catch(err =>
   this.set("status", `Biometric ID NOT OK: " + ${JSON.stringify(err)}`)
  );
 }

public doVerifyFingerprintWithCustomFallback(): void {
 this.fingerprintAuth
 .verifyFingerprintWithCustomFallback({
  message: "Skenirajte prst",
  fallbackMessage: "Unesite PIN",
  authenticationValidityDuration: 10
 })
 .then(() => this.set("status", "Biometric ID OK"))
 .catch(error => {
  this.set("status", "Biometric ID NOT OK: " + JSON.stringify(error));
  alert({
   title: "Biometric ID NOT OK",
   message: error.code === -3 ? "Show custom fallback" : error.message,
   okButtonText: "Okay"
  });
 });
}
}
```

U ovom dijelu imamo nekoliko funkcija: funkcija koja provjerava dostupnost (doCheckAvailable), koja provjerava je li se otisak prsta mijenjao (doCheckFingerprintsChanged). Također, imamo tri funckije koje verificiraju otisak prsta (doVerifyFingerprint, doVerifyFingerprintWithCustomUI, doVerifyFingerprintWithCustomFallback). Razlika među njima je koja se lozinka koristi kao fallback, defaultna ili custom napravljena.
Na početku file-a imamo klasu HelloWorldModel unutar koje imamo senzor FingerprintAuth.
Skener otiska prsta nastaje konstruktorom, a ostale vrijednosti se popunjavaju iz funkcija klase.

Izgled aplikacije



Literatura

- Materijal s vježbi iz kolegija “Mrežni i mobilni operacijski sustavi”
- Brad Martin - GitHub
- Eddy Verbruggen - GitHub
- Progress Telerik - YouTube
- Alex Ziskind - YouTube