|  |  |  |
| --- | --- | --- |
| **AI1** | Gajda Bartosz  Album 50970  Grupa 1 | **Wersja 1** |
|  |
| **LAB V** |

Cordova

# Spis treści

[Spis treści 1](#_Toc156321876)

[Cel zajęć 1](#_Toc156321877)

[Uwaga 1](#_Toc156321878)

[Instalacja Cordova 2](#_Toc156321879)

[Utworzenie projektu Cordova i pierwsze uruchomienie 2](#_Toc156321880)

[Podmiana projektu domyślnego na Pogodynkę 4](#_Toc156321881)

[Android 5](#_Toc156321882)

[Electron i aplikacja na Windows 8](#_Toc156321883)

[Debuggowanie aplikacji Electron 9](#_Toc156321884)

[Dostosowanie aplikacji Electron 10](#_Toc156321885)

[Budowa aplikacji 11](#_Toc156321886)

[Commit projektu do GIT 13](#_Toc156321887)

[Podsumowanie 13](#_Toc156321888)

# Cel zajęć

Celem głównym zajęć jest zdobycie następujących umiejętności:

* tworzenie hybrydowych aplikacji mobilnych z wykorzystaniem oprogramowania Apache Cordova;
* reużywanie istniejącego kodu HTML do tworzenia aplikacji mobilnych.

W praktycznym wymiarze uczestnicy zamienią swoją aplikację pogodową z LAB D na prostą aplikację mobilną Android.

# Uwaga

Ten dokument aktywnie wykorzystuje niestandardowe właściwości. Podobnie jak w LAB A wejdź do Plik -> Informacje -> Właściwości -> Właściwości zaawansowane -> Niestandardowe i zaktualizuj pola. Następnie uruchom ten dokument ponownie lub Ctrl+A -> F9.

# Instalacja Cordova

Upewnij się, że w systemie operacyjnym masz poprawnie zainstalowany lub rozpakowany i skonfigurowany menedżer pakietów NPM. Był on już wielokrotnie wykorzystywany, więc powinien być gotowy (por. LAB E, LAB F).

Otwórz ulubiony terminal i zainstaluj oprogramowanie Apache Cordova globalnie dla Twojej instancji NPM, poprzez wykonanie polecenia:

> npm install -g cordova

…

added 547 packages in 32s

# Utworzenie projektu Cordova i pierwsze uruchomienie

Utwórz i wejdź poprzez terminal do katalogu C:\Users\...\Desktop\ai1-cordova. Następnie utwórz swój projekt z wykorzystaniem polecenia:

> cordova create c50970

Creating a new cordova project.

Otwórz utworzony katalog c50970 w Visual Studio Code. Zapoznaj się ze strukturą katalogów.

Omów w maksymalnie 100 słowach zawartość projektu na tym etapie:

W utworzonym projekcie mamy następujące pliki

Index.css Index.js Index.html logo.png config.xml I package.json

…maksymalnie 100 słów…

Następnie wróć do terminala, wejdź do katalogu projektu i dodaj platformę browser, po czym uruchom emulację w przeglądarce:

> cd c50970

> cordova platform add browser

> cordova emulate browser

Powinna uruchomić się przeglądarka z domyślnym projektem:

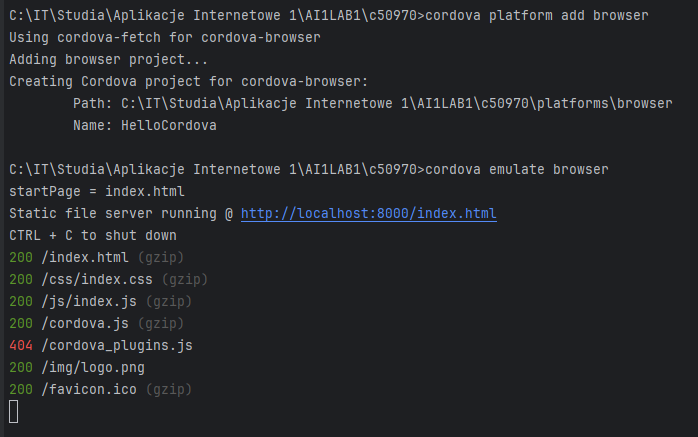
Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Wstaw zrzut ekranu emulacji w przeglądarce domyślnego projektu:



Wstaw zrzut ekranu zawartości terminalu po uruchomieniu emulacji i wyświetleniu strony:



Na jakim porcie uruchomiła się Tobie emulacja?

…numer portu = 8000

W maksymalnie 100 słowach opisz zmiany, które zaszły w strukturze katalogów projektu:

Dodały się moduły node oraz katalog platforms zawierający browser

Mamy też pliki package.json oraz package-lock.json

…maksymalnie 100 słów…

|  |  |  |
| --- | --- | --- |
| Punkty: | 0 | 1 |

# Podmiana projektu domyślnego na Pogodynkę

Zatrzymaj emulację w przeglądarce, jeśli jest jeszcze uruchomiona, z wykorzystaniem skrótu klawiszowego Ctrl + C.

Umieść w katalogu www projektu pliki HTML, JS i CSS z klientem REST do wyświetlania pogody, które opracowane zostały w ramach laboratorium LAB D. Upewnij się, że **główny plik HTML ma nazwę** weather.html. Upewnij się, że w pliku HTML w dalszym ciągu są poprawne ścieżki względne do plików JS i CSS. Upewnij się, że w kodzie użyty jest **Twój indywidualny klucz** do API.

Edytuj plik config.xml. Zmień domyślny plik projektu w znaczniku <content> z index.html na weather.html:

<content src="weather.html" />

Zmień także nazwę, opis i autora aplikacji w pliku config.xml.

Po wprowadzeniu zmian, zapisz wszystkie pliki i ponownie uruchom emulację projektu w przeglądarce:

> cordova emulate browser

Po uruchomieniu emulacji, w przeglądarce powinna pojawić się aplikacja Pogodynki. Wprowadź nazwę miejscowości i wyszukaj prognozy pogody, przykładowo:

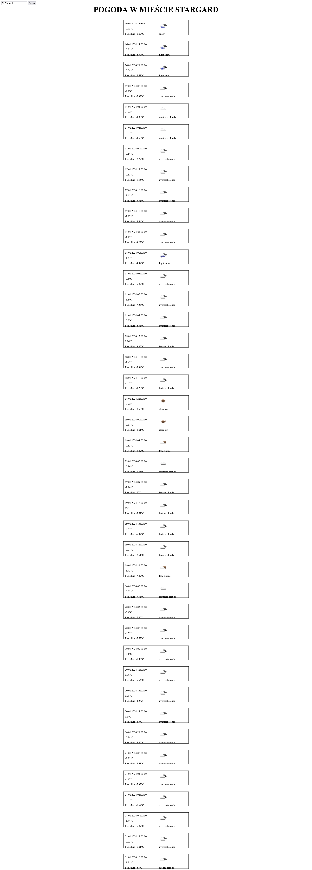
Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Ikona komputerowa

Opis wygenerowany automatycznie

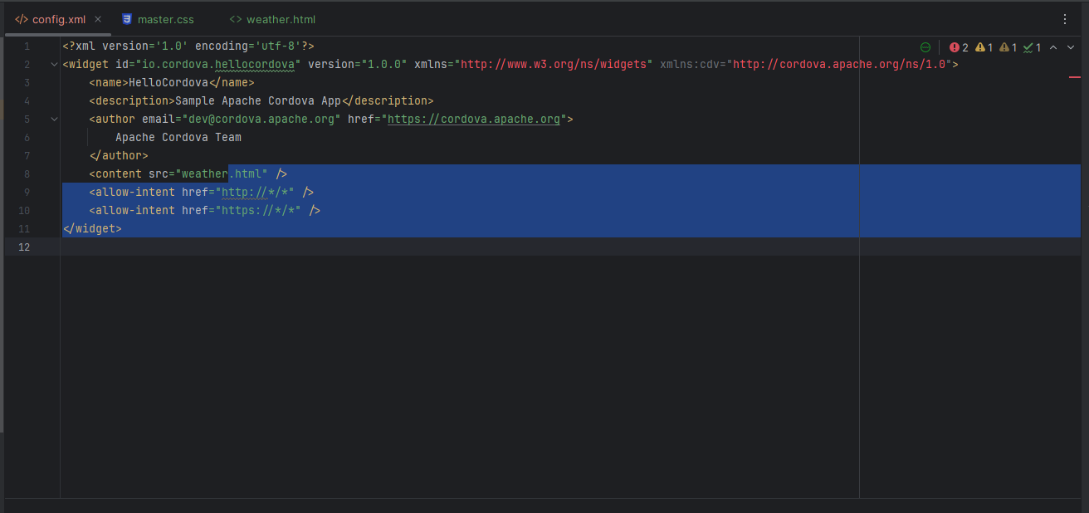
Wstaw zrzut ekranu emulacji w przeglądarce projektu Pogodynki przed wyszukaniem prognoz:



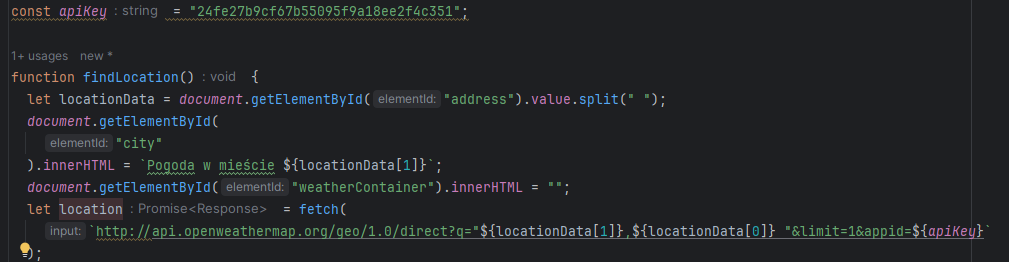
Wstaw zrzut ekranu emulacji pogodynki po wyszukaniu prognoz:



Wstaw zrzut ekranu pliku config.xml:



Wstaw zrzut ekranu fragmentu kodu zawierającego wykorzystanie Twojego klucza do API. Upewnij się, że klucz jest widoczny i że jest to Twój własny klucz:



|  |  |  |
| --- | --- | --- |
| Punkty: | 0 | 1 |

# Android

W tej sekcji wykorzystamy projekt Cordova do zbudowania aplikacji Pogodynka na Android. Do budowy wykorzystamy platformę VoltBuilder.

Najpierw w konsoli dodaj platformę Android:

> cordova platform add android

Następnie utwórz archiwum ZIP z całym katalogiem projektu, tj. C:\Users\...\Desktop\ai1-cordova\c50970. Załóżmy, że otrzymane archiwum to c50970.zip.

Wejdź na stronę <https://volt.build> . Zarejestruj i zaloguj się. Utworzone zostanie darmowe konto. Przejdź do sekcji Upload, tj. <https://volt.build/upload/> . Wybierz Android i wgraj swoje archiwum ZIP. Poczekaj, aż aplikacja zostanie zbudowana:

Obraz zawierający tekst, zrzut ekranu, diagram

Opis wygenerowany automatycznie

Po kilku minutach aplikacja będzie gotowa do pobrania:

Obraz zawierający tekst, zrzut ekranu, Czcionka, design

Opis wygenerowany automatycznie

Pobierz aplikację. Będzie to plik .apk.

Teraz jesteśmy gotowi do przetestowania działania aplikacji w emulatorze Android. Wykorzystamy do tego celu usługę appetize. Wejdź na strone <https://appetize.io/> . Przejdź do **Demo**. Przyciśnij **Upload your app** i wybierz z dysku swoją aplikację. Podaj adres email i kliknij **Generate Links**. Adres może być uczelniany lub prywatny. Testowane na @zut.edu.pl i @gmail.com. W ciągu kilku chwil otrzymasz na mail link do aplikacji:

Obraz zawierający tekst, zrzut ekranu, Czcionka

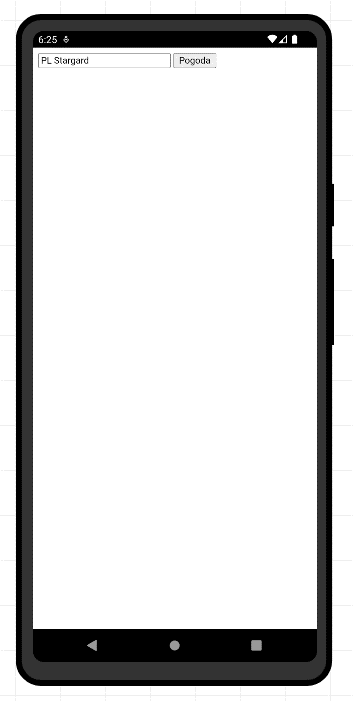
Opis wygenerowany automatycznie

Przejdź pod podany link w przeglądarce, wybierz urządzenie, system operacyjny i uruchom emulację:

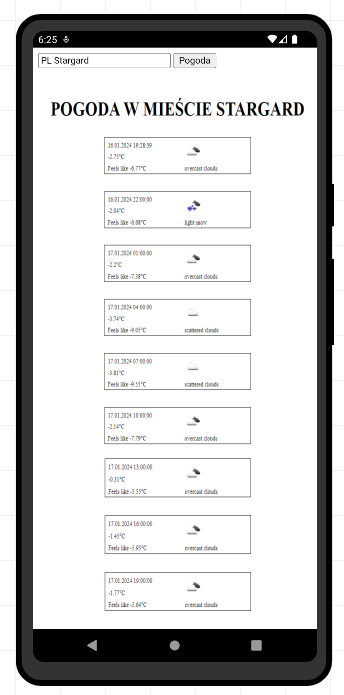
Obraz zawierający tekst, zrzut ekranu, numer

Opis wygenerowany automatycznie

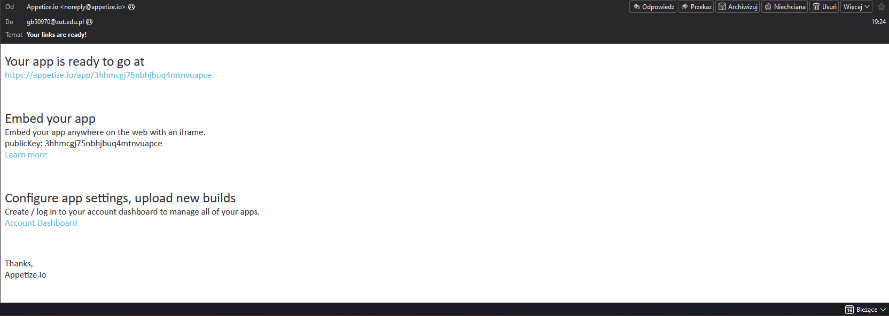
Wstaw zrzut ekranu Twojej aplikacji w **appetize** przed wyszukaniem prognozy:



Wstaw zrzut ekranu Twojej aplikacji w **appetize** po wyszukaniu prognozy:



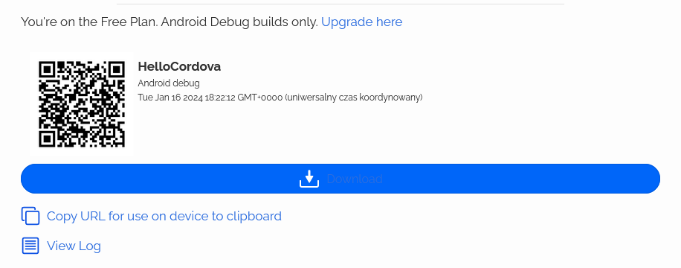
Wstaw zrzut ekranu maila od **appetize** z linkiem do aplikacji:



Wstaw link do aplikacji w **appetize**:

<https://appetize.io/app/3hhmcgj75nbhjbuq4mtnvuapce?device=pixel7&osVersion=13.0>

Wstaw zrzut ekranu fragmentu strony **VoltBuilder** przedstawiający zbudowaną aplikację:



|  |  |  |
| --- | --- | --- |
| Punkty: | 0 | 1 |

**Rada!**

Jeśli masz urządzenie Android, możesz zeskanować telefonem kod QR wyświetlony w **VoltBuilder**. Pod podanym adresem znajdziesz aplikację **.apk** do pobrania na Twój telefon. Następnie możesz zainstalować tę aplikację w telefonie i korzystać na co dzień!

# Electron i aplikacja na Windows

Dodaj do projektu platformę electron:

> cordova platform add electron

Przejdź do Visual Studio Code. W głównym katalogu projektu utwórz plik build.json o następującej zawartości:

{

"electron": {

"windows": {

"package": [

"zip"

]

}

}

}

Przy budowie aplikacji, biblioteka Electron tworzyć będzie archiwum ZIP z projektem, zamiast instalatora. Dostępne postaci pakietów instalacyjnych opisane są tutaj: <https://cordova.apache.org/docs/en/12.x/guide/platforms/electron/index.html#adding-a-package>.

Uruchom teraz emulację aplikacji Electron:

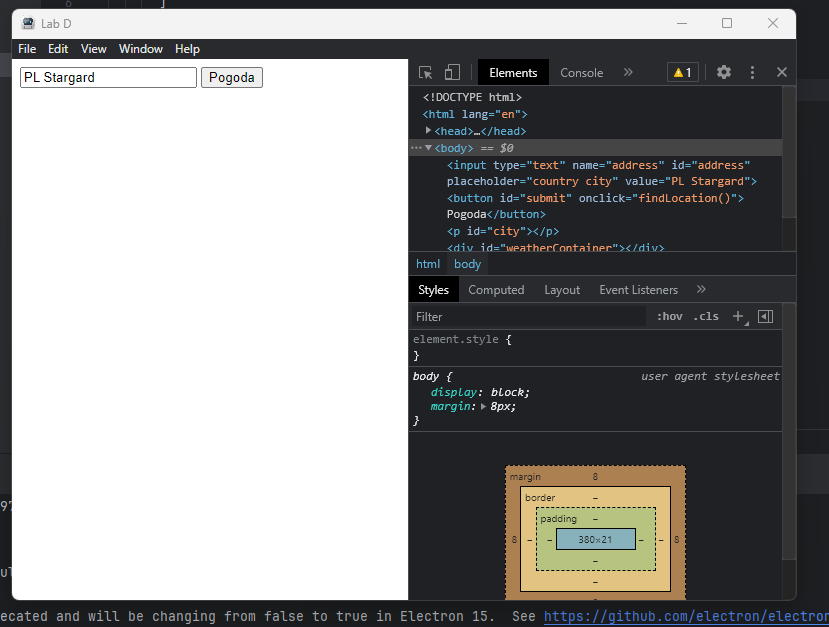
> cordova emulate electron –nobuild

W efekcie uruchomi się aplikacja Electron:

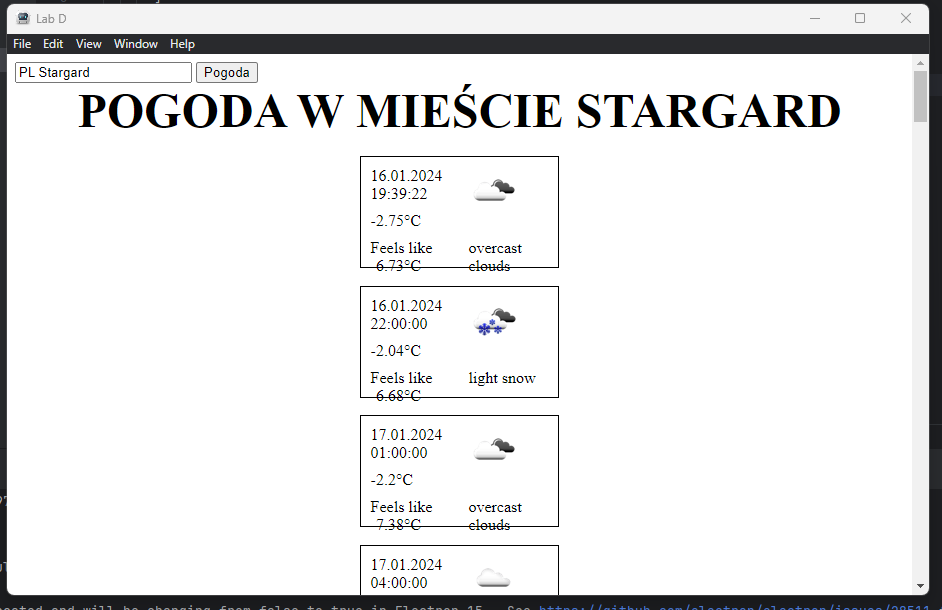
Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Wstaw zrzut ekranu uruchomionej aplikacji Electron:



Zamknij narzędzia deweloperske (panel z prawej strony) i wyszukaj prognozę pogody. Wstaw zrzut ekranu:

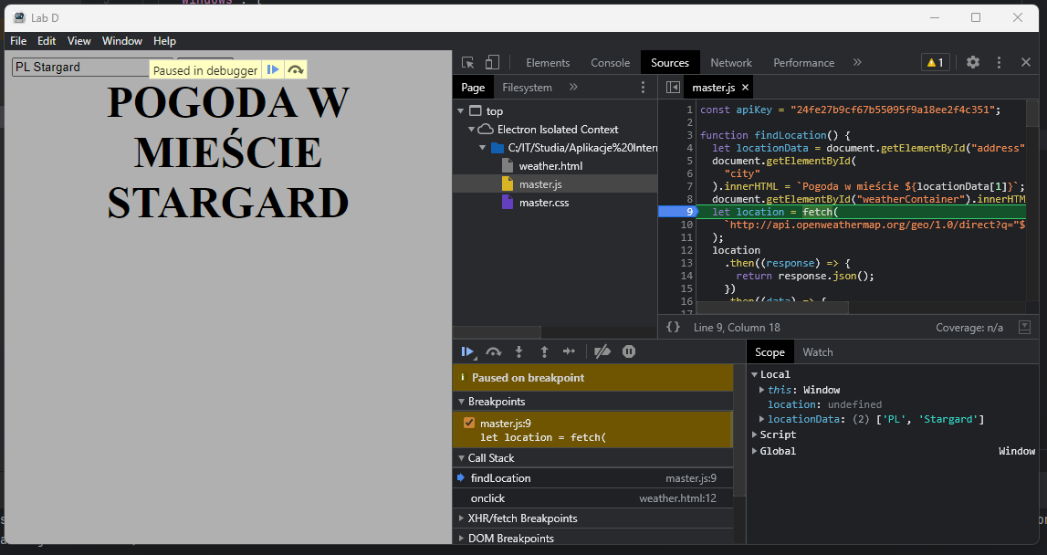


|  |  |  |
| --- | --- | --- |
| Punkty: | 0 | 1 |

# Debuggowanie aplikacji Electron

Ponownie włącz emulację alikacji electron. W narzędziach deweloperskich (panel z prawej strony), wybierz zakładkę „Sources”. Znajdź kod JS odpowiedzialny za pobranie bieżącej prognozy pogody i ustaw pułapkę (ang. breakpoint) na w linii odpowiedniej funkcji. Wyszukaj prognozę pogody. Upewnij się, że pułapka się aktywowała.

Wstaw zrzut ekranu całości okienka electron z aplikacją zatrzymaną na breakpoincie:



Upewnij się, że widoczne są:

- nad treścią aplikacji informacja, że wykonanie zostało wstrzymane w debuggerze

- debugger zatrzymany na pułapce (linia podświetlona na niebiesko)

- w okienku Scope widoczne są wartości zmiennych

|  |  |  |
| --- | --- | --- |
| Punkty: | 0 | 1 |

# Dostosowanie aplikacji Electron

W Visual Studio Code utwórz w projekcie podkatalog res\electron. Utwórz w dowolnym programie graficznym ikonkę aplikacji o wymiarach 512x512 pikseli (format PNG). Utworzony plik zapisz w katalogu res\electron, a następnie w pliku config.xml skonfiguruj ikonę dla platformy electron, przykładowo:

<platform name="electron">

<icon src="res/electron/cloud-sun.png" />

</platform>

Następnie utwórz plik res\electron\settings.json i zarejestruj go w config.xml:

<platform name="electron">

<icon src="res/electron/cloud-sun.png" />

<preference name="ElectronSettingsFilePath" value="res/electron/settings.json" />

</platform>

W pliku settings.json dostosuj szerokość i wysokość aplikacji, a także ukryj menu. Dokumentacja: <https://cordova.apache.org/docs/en/12.x/guide/platforms/electron/index.html#customizing-the-application's-window-options>

Przykładowa zawartość pliku settings.json:

{

"browserWindow": {

"width": 1024,

"height": 600,

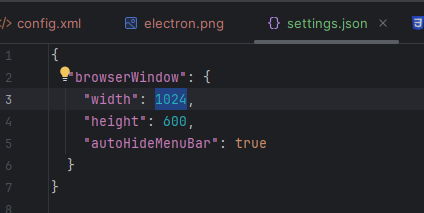
"autoHideMenuBar": true

}

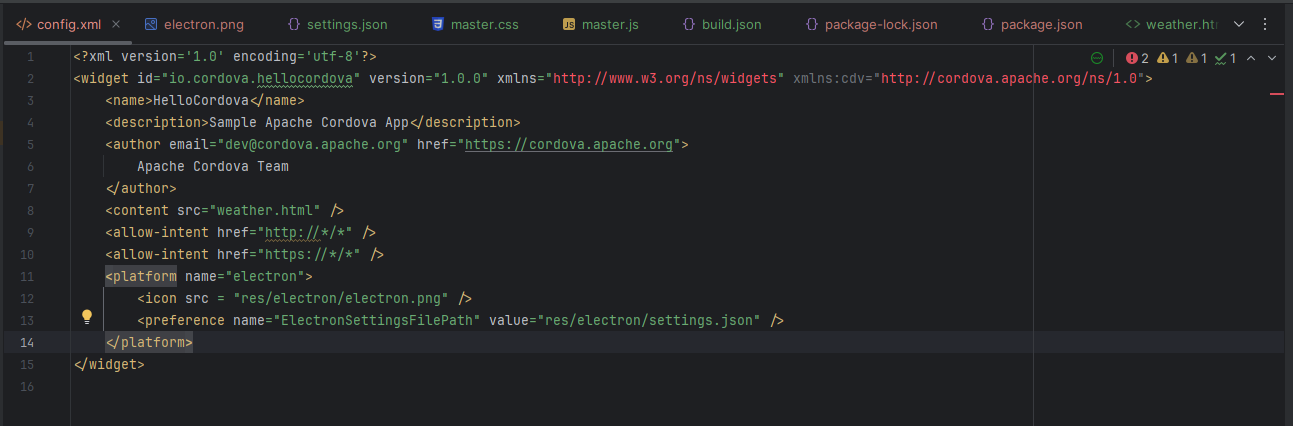
}

Ponownie dokonaj emulacji aplikacji, tym razem z przełącznikiem --release, który ukryje narzędzia deweloperskie.

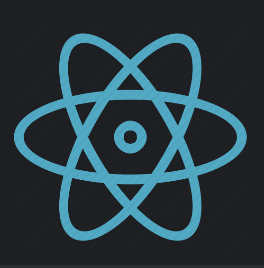
Wstaw zrzut ekranu kodu pliku settings.json:



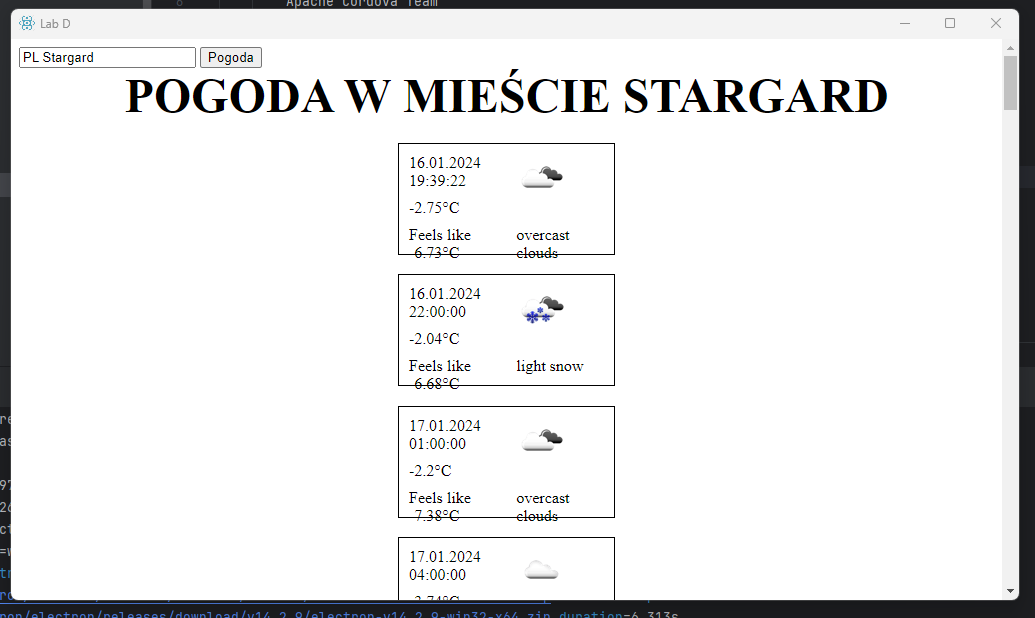
Wstaw zrzut ekranu kodu pliku config.xml:



Wstaw zrzut ekranu pliku PNG z ikoną o wymiarach 512x512px:



Wstaw zrzut ekranu uruchomionej aplikacji. Upewnij się, że wysokość i szerokość odpowiada wartościom z konfiguracji, pasek menu jest ukryty, ikonka jest zgodna ze skonfigurowaną, a narzędzia deweloperskie są zamknięte:



|  |  |  |
| --- | --- | --- |
| Punkty: | 0 | 1 |

# Budowa aplikacji

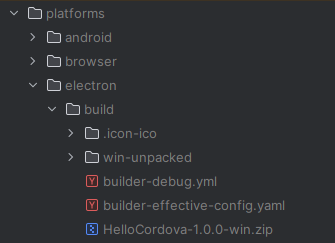
Wykonaj polecenie budowy aplikacji Electron:

> cordova build electron --release

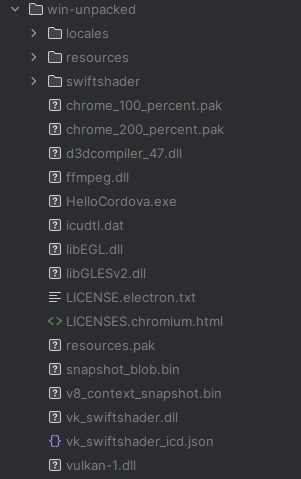
Odczekaj kilka minut. Po zakończonym procesie wynikowe archiwum ZIP znajdzie się w katalogu \platforms\electron\build.

Skopiuj archiwum do innego katalogu, rozpakuj, wejdź do środka rozpakowanego katalogu i uruchom plik EXE.

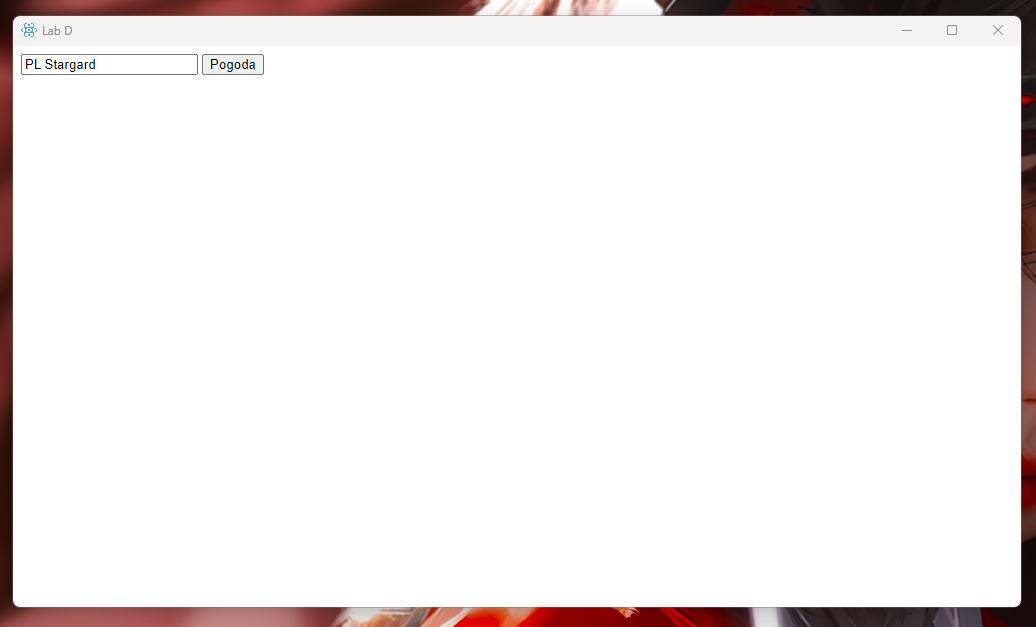
Wstaw zrzut ekranu zawartości katalogu platforms\electron\build:



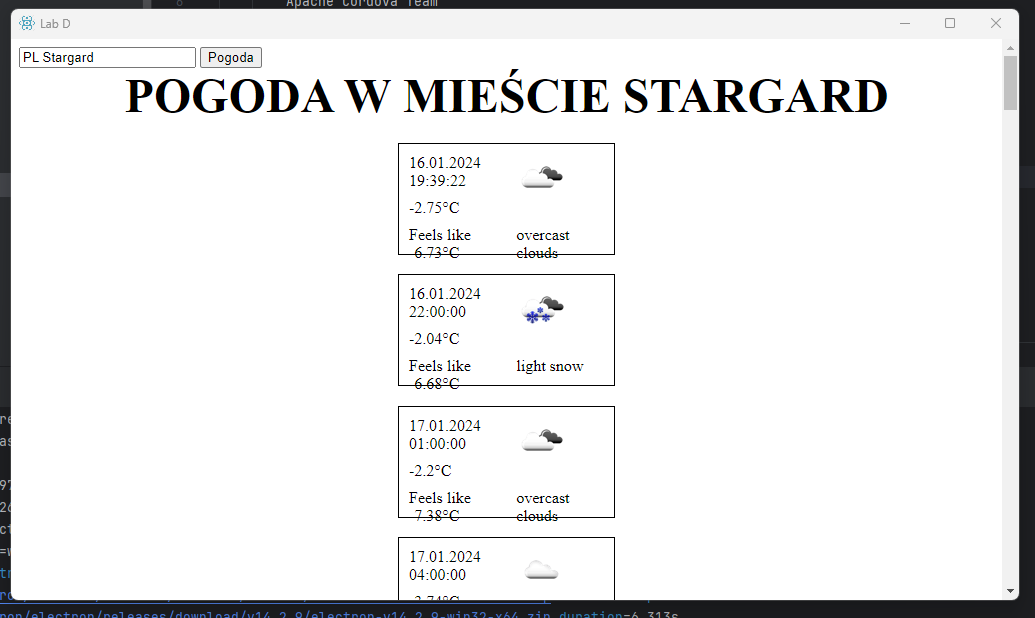
Wstaw zrzut ekranu zawartości rozpakowanego archiwum ZIP z aplikacją:



Wstaw zrzut ekranu uruchomionej aplikacji przed sprawdzeniem pogody:



Wstaw zrzut ekranu uruchomionej aplikacji po sprawdzeniu prognozy pogody:



Wstaw zrzut ekranu eksploratora plików z zaznaczonym plikiem EXE aplikacji z ikonką zgodną z konfiguracją:



|  |  |  |
| --- | --- | --- |
| Punkty: | 0 | 1 |

# Commit projektu do GIT

Zacommituj i pushnij swoje rozwiązanie do **swojego** repozytorium GIT. **UWAGA! Nie commitować binarek!!!**

Upewnij się, czy wszystko dobrze się wysłało. Jeśli tak, to z poziomu przeglądarki utwórz branch o nazwie lab-v na podstawie głównej gałęzi kodu.

Podaj link do brancha lab-v w swoim repozytorium:

…link, np. https://github.com/1Batrex1/ai1-lab1/tree/lab-v

# Podsumowanie

W kilku zdaniach podsumuj zdobyte podczas tego laboratorium umiejętności.

Nauczyłem się że w sumie cordova to fajne środowisko jeśli chce się wydać aplikację na wiele platform

…podsumowanie…

Zweryfikuj kompletność sprawozdania. Utwórz PDF i wyślij w terminie.