CS 528
Ubiquitous and Mobile Computing
Tech Talk: Ionic

Yijie Yan, Qinlun Luan,
Wei Xiong, Zinan Yue
Computer Science Dept.
WPI
Background

Native

Application developed specifically for a mobile operating system.

Hybrid

Hybrid applications are, at core, websites packaged into a native wrapper.

VS
Background

Native app

Web service (database)

HTTP request

Mobile website

Web server

HTTP request

Hybrid app

Web service (database)

HTTP request

This is where the app code exists.

Native App

SDKs

Platform

This is where the app code exists.

Browser

Platform

This is where the app code exists.

WebView

Native app

Cordova

Platform
Background

Ionic is an APP development platform

Help developers build and deploy cross-platform apps. From open source to premium services, Ionic makes app creation lightning fast.

An open source UI toolkit for building mobile and desktop apps using web technologies (HTML, CSS, and JavaScript).

Focused on frontend user experience, or UI interaction of an app (controls, interactions, gestures, animations).

Has official integrations with Angular and React, and support for Vue is in development.
## Specific Problems

<table>
<thead>
<tr>
<th>Cross-platform</th>
<th>Web Standards-based</th>
<th>Beautiful Design</th>
<th>Simplicity to use</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>One code base</strong></td>
<td><strong>Web tech</strong></td>
<td><strong>Base theme</strong></td>
<td><strong>Easy develop</strong></td>
</tr>
<tr>
<td>Build and deploy apps across <strong>multiple platforms</strong>, such as native iOS, Android, desktop, and the web as a Progressive Web App</td>
<td>Built on top of reliable, standardized <strong>web technologies</strong>: HTML, CSS, and JavaScript</td>
<td>Designed to work and display cross all platforms. Start with pre-designed components, typography, interactive paradigms, and a base theme.</td>
<td>Creating Ionic apps is easy to learn, and accessible to anyone with <strong>web development</strong> skills.</td>
</tr>
</tbody>
</table>
## Use Case

<table>
<thead>
<tr>
<th>Device</th>
<th>Device loads Cordova app wrapper</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cordova APP Wrapper</th>
<th>WebView with JavaScript API</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>APACHE CORDOVA</strong></td>
<td></td>
</tr>
</tbody>
</table>

Cordova is a hybrid mobile app platform for building mobile apps that can run using HTML, CSS, and JavaScript inside of a native app. It’s a utility for creating a bridge between the platform and the application.

<table>
<thead>
<tr>
<th>Web app</th>
<th>Angular</th>
<th>Ionic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Angular is a web app that controls the app routing and function.

Ionic provides the user interface components rendered in the app.
How it Works

User opens the app

Device
- Device loads Cordova app wrapper
- Device opens camera app
- Camera app closes and returns photo

APACHE CORDOVA
- Cordova app wrapper loads new WebView
- Cordova asks device to open camera app
- Cordova receives and passes on photo

WebView with JavaScript API
- WebView loads index.html file

WebKit with JavaScript API
- Cordova asks JavaScript API
- Cordova JavaScript API asks for camera
- Cordova JavaScript API passes on photo
- Angular gets photo and returns the view

Web app
- Angular bootstraps and determines default view
- Ionic components are rendered for the UI

Angular
- Angular calls Cordova camera JavaScript API
- Ionic button is passed, calls Angular

Ionic
- Ionic components are rendered for the UI
Real World Examples

Amtrak did a complete overhaul of their existing app and converted it to Ionic.

GE Transportation uses Ionic to build mission critical apps ensuring timely deliveries.

The National Health Service, one of the UK’s top health providers, went all in on Ionic.

The Marketwatch team built their flagship app in record time leveraging Ionic.
**Code Snippet**

`/src/index.html`

```html
<!-- Ionic's root component and where the app will load -->
<ion-app></ion-app>

<!-- The polyfills js is generated during the build process -->
<script src="build/polyfills.js"></script>

<!-- The vendor js is generated during the build process
     It contains all of the dependencies in node_modules -->
<script src="build/vendor.js"></script>

<!-- The main bundle js is generated during the build process -->
<script src="build/main.js"></script>
```

`/src/index.html` is the main entry point for the app, though its purpose is to set up scripts, CSS includes, and bootstrap, or start running our app.

Ionic looks for the `<ion-app>` tag in your HTML.
@NgModule({
  declarations: [MyApp,
  HelloIonicPage, ItemDetailsPage,
  ListPage],
  imports: [BrowserModule,
  IonicModule.forRoot(MyApp)],
  bootstrap: [IonicApp],
  entryComponents: [MyApp,
  HelloIonicPage, ItemDetailsPage,
  ListPage],
  providers: [StatusBar, SplashScreen,
  {provide: ErrorHandler, useClass: IonicErrorHandler}]
})
export class AppModule {}
we set up an `ion-menu` to function as a side menu, and then an `ion-nav` component to act as the main content area. The `ion-menu`'s `[content]` property is bound to the local variable `content` from our `ion-nav`, so it knows where it should animate around.
<ion-header>
  <ion-navbar>
    <button menuToggle *ngIf="!selectedItem">
      <ion-icon name="menu"></ion-icon>
    </button>
    <ion-title>Item Details</ion-title>
  </ion-navbar>
</ion-header>

<ion-content>
  <h3 text-center *ngIf="selectedItem">
    {{selectedItem.title}}
    <ion-icon [name]="selectedItem.icon"></ion-icon>
  </h3>
  <h4 text-center *ngIf="selectedItem">
    You navigated here from
    <b>{{selectedItem.title}}</b>
  </h4>
</ion-content>