



دانشکده مهندسی کامپیوتر

# طراحی و پیاده‌سازی شبکه اجتماعی محلی و بی‌سیم بدون اینترنت

پروژه برای دریافت درجه کارشناسی در رشته مهندسی کامپیوتر  
گرایش هوش مصنوعی و رباتیک

مریم سادات هاشمی

استاد راهنما

سید صالح اعتمادی

آبان ۱۳۹۸



## تأییدیه‌ی صحت و اصالت نتایج

باسمه تعالی

اینجانب مریم سادات هاشمی به شماره دانشجویی ۹۴۵۲۳۲۵۲ دانشجوی رشته مهندسی کامپیوتر مقطع تحصیلی کارشناسی تأیید می‌نمایم که کلیه‌ی نتایج این پروژه حاصل کار اینجانب و بدون هرگونه دخل و تصرف است و موارد نسخه‌برداری شده از آثار دیگران را با ذکر کامل مشخصات منبع ذکر کرده‌ام. در صورت اثبات خلاف مندرجات فوق، به تشخیص دانشگاه مطابق با ضوابط و مقررات حاکم ( قانون حمایت از حقوق مؤلفان و مصنفان و قانون ترجمه و تکثیر کتب و نشریات و آثار صوتی، ضوابط و مقررات آموزشی، پژوهشی و انضباطی ... ) با اینجانب رفتار خواهد شد و حق هرگونه اعتراض درخصوص احقاق حقوق مکتسب و تشخیص و تعیین تخلف و مجازات را از خویش سلب می‌نمایم. در ضمن، مسؤولیت هرگونه پاسخگویی به اشخاص اعم از حقیقی و حقوقی و مراجع ذی‌صلاح (اعم از اداری و قضایی) به عهده‌ی اینجانب خواهد بود و دانشگاه هیچ‌گونه مسؤولیتی در این خصوص نخواهد داشت.

نام و نام خانوادگی: مریم سادات هاشمی

تاریخ و امضا:

## مجوز بهره‌برداری از پایان‌نامه

بهره‌برداری از این پایان‌نامه در چهارچوب مقررات کتابخانه و با توجه به محدودیتی که توسط استاد راهنما به شرح زیر تعیین می‌شود، بلامانع است:

- ☐ بهره‌برداری از این پایان‌نامه برای همگان بلامانع است.
- ☐ بهره‌برداری از این پایان‌نامه با اخذ مجوز از استاد راهنما، بلامانع است.
- ☐ بهره‌برداری از این پایان‌نامه تا تاریخ ..... ممنوع است.

استاد راهنما: سید صالح اعتمادی

تاریخ:

امضا:

## قدردانی

سپاس خداوندگار حکیم را که با لطف بی‌کران خود، آدمی را زیور عقل آراست. در آغاز وظیفه خود می‌دانم از زحمات بی‌دریغ استاد راهنمای خود، جناب آقای دکتر سید صالح اعتمادی، صمیمانه تشکر و قدردانی کنم که قطعاً بدون راهنمایی‌های ارزنده ایشان، این مجموعه به انجام نمی‌رسید. همچنین لازم می‌دانم از پدید آورندگان بسته زی‌پرشین، مخصوصاً جناب آقای وفا خلیقی، که این پایان‌نامه با استفاده از این بسته، آماده شده است و همه دوستانمان در گروه پارسی‌لاتک کمال قدردانی را داشته باشم. در پایان، بوسه می‌زنم بر دستان خداوندگاران مهر و مهربانی، پدر و مادر عزیزم و بعد از خدا، ستایش می‌کنم وجود مقدس‌شان را و تشکر می‌کنم از خانواده عزیزم به پاس عاطفه سرشار و گرمای امیدبخش وجودشان، که بهترین پشتیبان من بودند.

مریم سادات هاشمی

آبان ۱۳۹۸

## چکیده

هدف این پروژه اضافه کردن لایه خدمات شبکه اجتماعی به یک شبکه ارتباطی بی سیم مش می باشد. در این پروژه ابتدا یک اپلیکیشن موبایل برای ارائه سرویس همسایگی پیاده سازی می شود. سپس خدمات لازم از لایه خدمات اجتماعی برای این سرویس شناسایی می شوند. سپس خدمات این لایه برای ارائه روی شبکه ارتباطی بی سیم مش بازمهندسی، طراحی و پیاده سازی می شوند. در نهایت با استفاده از این اپلیکیشن می توانیم شبکه ی اجتماعی را راه اندازی کنیم که حتی بدون اتصال به اینترنت قادر به ارائه خدمات اجتماعی برخط مثل ارسال پیغام های شخصی یا عمومی می باشد.

واژگان کلیدی: شبکه اجتماعی، بی سیم، شبکه مش، اپلیکیشن موبایل، همسایگی

# فهرست مطالب

خ	فهرست تصاویر
۱	فصل ۱: مقدمه
۴	۱-۱ کارهای مربوطه
۴	۱-۱-۱ Open Garden
۴	۱-۱-۲ NextDoor
۴	۱-۱-۳ Fire Chat
۶	فصل ۲: انتخاب API مناسب برای ایجاد ارتباط Peer to Peer
۷	۲-۱ فناوری WiFi Direct
۷	۲-۲ بررسی فنی
۷	۲-۳ معماری
۸	۲-۴ تشکیل گروه
۹	۲-۴-۱ استاندارد
۹	۲-۴-۲ مستقل
۱۰	۲-۴-۳ پایدار
۱۰	۲-۵ امنیت
۱۰	۲-۶ ذخیره انرژی
۱۰	۲-۶-۱ Opportunistic Power Save protocol
۱۱	۲-۶-۲ Notice of Absence (NOA) protocol

۷-۲ فواید ۱۱ . . . . .

۱۲ فصل ۳: طراحی و پیاده‌سازی

۱-۳ اتصال دستگاه‌ها در لایه‌ی فیزیکی با استفاده از فناوری WifiDirect ۱۲ . . . . .

۱-۳-۱ Discovery ۱۲ . . . . .

۱-۳-۲ Connect ۱۳ . . . . .

۱-۳-۳ Create Group ۱۳ . . . . .

۲-۳ دریافت و ارسال پیام ۱۳ . . . . .

۱-۲-۳ Multi thread یا Synchronous ۱۴ . . . . .

۲-۲-۳ Asynchronous ۱۶ . . . . .

۱۹ مراجع



## فهرست تصاویر

۸ . . . . . ۱-۲ معماری Wifi Direct [۱]

# فصل ۱

## مقدمه

یکی از نیازهای اساسی انسان‌ها نیاز به برقراری ارتباط با دیگران و اجتماعی شدن است. پایه‌ای‌ترین ارتباطات اجتماعی را می‌توان خانواده، بستگان، دوستان و همکاران دانست که روابط معناداری بین افراد در این گروه‌ها برقرار است. این در حالی است که در سال‌های اخیر فناوری دیجیتال و اینترنت شکل جدیدی از ارتباط را تعریف کرده‌اند و در حال حاضر شاهد گسترش بی سابقه شبکه‌های اجتماعی<sup>۱</sup> در بین مردم هستیم. به گونه‌ای که اکنون شبکه‌های اجتماعی جزئی از زندگی روزمره مردم شده‌است.

شبکه‌های اجتماعی به کاربران این فرصت را می‌دهد تا آراء و نظرات خود را با دیگران مطرح کنند و از نظرات مختلف مطلع شوند. در سایت‌های خرید و فروش و کالای موردنظر خود را جستجو کنند و نظرات خریداران قبلی را بخوانند و سفارش خود را ثبت کنند. به راحتی اطلاعات مقصد و هر آنچه برای یک سفر لازم است را از اینترنت دریافت کرده و از نظرات و راهنمایی‌های افرادی که قبلاً این سفر را تجربه کرده‌اند؛ استفاده کنند و با طیب خاطر به سفر بروند. در فعالیتهای گروهی و تشکلهای مردم‌نهاد مشارکت کنند. بسیاری از فعالیتهای این چنینی که ذاتی اجتماعی دارند با کمک اینترنت و ظهور شبکه‌های اجتماعی برای کاربران آسان شده‌اند.

در بین همه‌ی این قابلیت‌ها که شبکه‌های اجتماعی کنونی در اختیار ما قرار می‌دهند، شرایطی وجود دارد که این شبکه‌های اجتماعی پاسخگوی نیازهای ما نیستند. مثلاً بخواهید خبر گم شدن حیوان خانگی‌تان را در محله‌ی خود اعلام کنید و یا بخواهید یک نردبان قرض بگیرید و یا به دنبال یک دندان‌پزشکی یا لوله‌کش خوب

---

<sup>1</sup> Social Networks

در نزدیکی محل زندگیتان باشید. تمامی این مثال‌ها نیازمند این است که شما با همسایگانان و افرادی که در نزدیکی شما و محله‌تان زندگی می‌کنند؛ در ارتباط باشید. موضوعی که امروزه در زندگی ما کمرنگ شده‌است. از طرفی روند شبکه‌های اجتماعی مثل فیسبوک<sup>۲</sup>، توییتر<sup>۳</sup> و ... به گونه‌ای است که ما را با دوستانی که ۲۰ سال گذشته با آن‌ها ارتباط داشتیم، متصل می‌کند اما با افرادی که در نزدیکی ما زندگی می‌کنند و نیاز بیشتری به ارتباط با آن‌ها داریم، متصل نمی‌کند.

بنابراین در این پروژه ما قصد داریم که یک شبکه‌ی اجتماعی محلی را پیاده‌سازی کنیم که امکان برقراری ارتباط با همسایگانمان را برای ما ایجاد کند. از طرفی چون هدف ما در این شبکه‌ی اجتماعی اتصال به افرادی است که از نظر بعد جغرافیایی به ما نزدیک هستند، می‌توانیم با استفاده از فناوری‌هایی همچون بلوتوث<sup>۴</sup>، اتصال نقطه به نقطه وای فای<sup>۵</sup> و ... این ارتباط را حاصل کنیم بدون این که بخواهیم از شبکه‌ی جهانی اینترنت<sup>۶</sup> استفاده کنیم. برای رسیدن به این منظور از شبکه‌ی بی‌سیم مش<sup>۷</sup> نیز استفاده خواهیم کرد. البته لازم به ذکر است که در حالت ایده آل اگر تعداد کاربران این شبکه‌ی اجتماعی مقدار قابل توجهی در سراسر جهان باشد، می‌توان با کاربران دور دست نیز ارتباط برقرار کرد. سایر کاربردهای این شبکه‌ی اجتماعی به صورت زیر خواهد بود:

۱. فرض کنید که شما در نزدیکی دانشکده‌ی مهندسی کامپیوتر قرار دارید و یک رویداد در دانشکده‌ی مهندسی کامپیوتر در حال برگزاری است. بنابراین اپلیکیشن به صورت یک اعلان بر روی گوشی شما، برگزاری رویداد را به شما اطلاع رسانی می‌کند.

۲. فرض کنید شما برای روز چهارشنبه، نیاز به کمک کسی دارید که از فرزندان نگهداری کند. شما می‌توانید درخواست یک پرستار بچه را در این شبکه محلی به اشتراک بگذارید.

۳. شما دنبال یک کارواش، مکانیکی یا دندان پزشکی خوب در نزدیکی خانه‌تان هستید. می‌توانید در مورد این موضوعات از دیگران در این شبکه‌ی محلی بپرسید.

۴. فرض کنید که شما دوچرخه‌تان را اطراف خانه‌تان گم کرده‌اید. می‌توانید این موضوع را در شبکه محلی

<sup>۲</sup>Facebook

<sup>۳</sup>Twitter

<sup>۴</sup>Bluetooth

<sup>۵</sup>Peer to Peer WiFi Connection

<sup>۶</sup>Internet

<sup>۷</sup>Wireless Mesh Network

اعلان کنید تا اگر پیدا شد دیگران به شما در این شبکه‌ی محلی خبر بدهند یا مثلاً می‌توانید این موضوع را به عنوان جرم در یک منطقه یا امن نبودن یک محله اعلام کنید.

۵. نهادهای دولتی می‌توانند در این شبکه عضو بشوند و ساکنین یک محله را از اتفاقاتی مثل آتش‌سوزی، دزدی، خرابی تلفن، قطع آب و گاز و برق و ... مطلع کنند و یا برعکس به این شکل که در صورت اتفاق افتادن هر یک از این حوادث ساکنین آن محله بتوانند در سریع‌ترین زمان ممکن نهاد دولتی مربوطه را مطلع سازند.

۶. فرض کنید که یک سازمان مثل هواشناسی در این شبکه محلی عضو شود و مردم یک شهر، روستا یا یک استان را از آب و هوای آن روز با خبر کند. مثلاً امروز در مناطق جنوبی شهر آب گرفتگی داریم یا امروز هوا در ساعات ۲ تا ۳ بعد از ظهر بارانی است؛ لطفاً از چتر استفاده کنید.

۷. شما می‌توانید زمان رفتن به سر کار خود را به بقیه اعلام کنید و اگر کسی از همسایگان‌تان در مسیر شما قرار داشت، با شما همراه شود. (کاهش ترافیک و آلودگی هوا)

۸. فروشگاه‌ها و هایپرمارکت‌های محله می‌توانند موجودی کالاهای خود را اعلام کنند و یا تخفیف‌های اقلام مختلف را در این شبکه‌ی محلی قرار بدهند.

۹. می‌توانید از همسایگان‌تان در خواست اجاره‌ی حیاط، پارکینگ یا خانه‌شان را برای برگزاری جلسه‌ها و مراسم‌هایتان و ... داشته باشید.

۱۰. فرض کنید شما نیاز دارید که بسته‌ای را در نزدیکی محدوده‌ی زندگی‌تان ارسال کنید ولی وقت کافی ندارید و می‌توانید در شبکه درخواست کنید که یک نفر به صورت رایگان این کار را برای شما انجام دهد.

۱۱. می‌توانید در مورد مسائل مختلف محله‌تان رای‌گیری کنید مثل تغییر نام یک کوچه.

## ۱-۱ کارهای مربوطه

## Open Garden ۱-۱-۱

**Open Garden**<sup>۸</sup> یک سرویس است که این امکان را به مردم می‌دهد که خدمات اینترنت خود را با افرادی که در نزدیکی آنها هستند؛ به اشتراک بگذارند. بنابراین با استفاده از این دستگاه اشتراک‌گذاری اینترنت، هر کس می‌تواند پهنای باند اضافی اینترنت خود را ارائه دهد و برای آن پول بگیرد یا خدمات اینترنتی را از دیگران خریداری کند. [۱۰]

## NextDoor ۲-۱-۱

**NextDoor**<sup>۹</sup> یک سرویس شبکه اجتماعی خصوصی برای محله است. این شرکت در سال ۲۰۰۸ در سانفرانسیسکو، کالیفرنیا تأسیس شد و در اکتبر ۲۰۱۱ در ایالات متحده راه اندازی شد. کاربران NextDoor نام و نشانی واقعی خود را به وب سایت ارائه می‌دهند. پیام‌های منتشر شده در وب سایت، فقط برای سایر اعضای NextDoor در همان محله قابل مشاهده است. [۹]

## Fire Chat ۳-۱-۱

**FireChat**<sup>۱۰</sup> یک برنامه اختصاصی تلفن همراه است که توسط Open Garden توسعه داده شده است. این شبکه‌ی اجتماعی از شبکه بی‌سیم مش استفاده می‌کند تا تلفن‌های هوشمند بتوانند از طریق بلوتوث، وای فای یا در چارچوب Multipeer شرکت اپل بدون اتصال به اینترنت به یکدیگر متصل شوند. همچنین این برنامه این قابلیت را دارد که از طریق اینترنت نیز اتصال برقرار شود.

FireChat به عنوان یک ابزار ارتباطی در برخی اعتراضات مدنی مورد استفاده قرار گرفت؛ اگر چه که برای چنین اهدافی طراحی نشده است. از جمله دلایلی که این شبکه‌ی اجتماعی محبوبیت چندانی بین مردم ندارد؛ می‌توان به ضعف در پیاده‌سازی، عدم اتصال مناسب بین دو دستگاه اندروید<sup>۱۱</sup> و آیفون<sup>۱۲</sup>، اشکال در عدم

<sup>۸</sup><https://www.opengarden.com>

<sup>۹</sup><https://nextdoor.com/>

<sup>۱۰</sup><https://www.opengarden.com/firechat/>

<sup>۱۱</sup>Android

<sup>۱۲</sup>iphone

ارسال مناسب عکس و غیره اشاره نمود. [۸]

## فصل ۲

### انتخاب API مناسب برای ایجاد ارتباط Peer to

### Peer

اولین گام برای ساخت یک شبکه‌ی محلی این است که یک API موجود یا فناوری موجود را که ارتباط محلی و Peer to Peer را برای ما بوجود بیاورد را پیدا کنیم و یا این که از ابتدا قابلیت‌های مورد نیاز را پیاده‌سازی کنیم. بدین منظور چندین API شامل p2pkit [۶]، Alljoyn [۲]، Nearby [۴]، Hypelabs [۳] و Open Garden [۵] را پیدا کردیم. با این که هر کدام از این API ها قابلیت‌های مورد نیاز و اولیه‌ی شبکه‌های اجتماعی را در بردارند، اما مشکل اساسی همه‌ی این API ها این است که اپلیکیشنی که بر مبنای این‌ها پیاده‌سازی شده‌باشد، باید به اینترنت متصل باشد تا API بتواند به درستی کار کند. بنابراین از آن جایی که شبکه‌ی اجتماعی ما قرار است بدون اینترنت کار کند، نمی‌توانیم از این API ها بهره ببریم.

در ادامه، با فناوری WiFi Direct آشنا می‌شویم که بر روی گوشی‌های اندروید موجود است و تمامی قابلیت‌هایی که ما برای برقراری ارتباط محلی گوشی‌ها به یکدیگر را نیاز داریم؛ در خود دارد. با این تفاوت که مشکل API ها را هم ندارد و بدون نیاز به اتصال اینترنت می‌تواند کار کند. در ادامه‌ی این فصل به معرفی این فناوری و قابلیت و مزایای آن می‌پردازیم.

## ۲-۱ فناوری WiFi Direct

WiFi Direct فناوری جدید تعریف شده توسط اتحادیه WiFi<sup>۱</sup> است که هدف آن ارتقاء ارتباط مستقیم بین دستگاه‌ها است؛ بدون این که به یک نقطه دسترسی بی‌سیم<sup>۲</sup> نیاز باشد. WiFi Direct بر روی زیر ساخت موفق IEEE 802.11 بنا شده است و این اجازه را به دستگاه‌ها می‌دهد که در یک ارتباط، یک دستگاه نقش نقطه دسترسی بی‌سیم را ایفا کند و عملکرد آن را انجام دهد. در حال حاضر می‌توان با استفاده از استاندارد IEEE 802.11 یک ارتباط مستقیم بین دستگاه‌ها ایجاد کرد. اما اشکالات فراوانی همچون مصرف زیاد انرژی در دستگاه وجود دارد. [۷]

## ۲-۲ بررسی فنی

در یک شبکه معمولی WiFi مشتری<sup>۳</sup> اسکن می‌کند و عضو یکی از شبکه‌های بی‌سیم موجود که توسط نقطه دسترسی بی‌سیم ایجاد و اعلام شده است؛ می‌شود. این فرایند در WiFi Direct به صورت پویا<sup>۴</sup> انجام می‌شود از این رو یک دستگاه WiFi Direct باید هر دو نقش مشتری و نقطه دسترسی بی‌سیم را به طور همزمان اجرا کند. [۷]

## ۲-۳ معماری

دستگاه‌های دارای WiFi Direct با ایجاد یک گروه با عنوان P2P Group می‌توانند با یکدیگر ارتباط برقرار کنند. دستگاهی که عملکردی همچون نقطه دسترسی بی‌سیم دارد را P2P Group Owner می‌نامند و دستگاهی که در نقش مشتری است را P2P client گویند. هنگامی که یک P2P Group ایجاد می‌شود، سایر مشتری‌ها می‌توانند با همان روش سنتی شبکه‌های WiFi به گروه بپیوندند. زمانی که یک دستگاه هم در نقش P2P Client و هم در نقش P2P Group Owner باشد، دستگاه به طور متناوب با استفاده از اشتراک زمانی<sup>۵</sup> بین این دو نقش تغییر می‌کند. (مثال: لی‌تاپ ۲ در بالای شکل ۲-۱) [۷]

<sup>1</sup> WiFi alliance

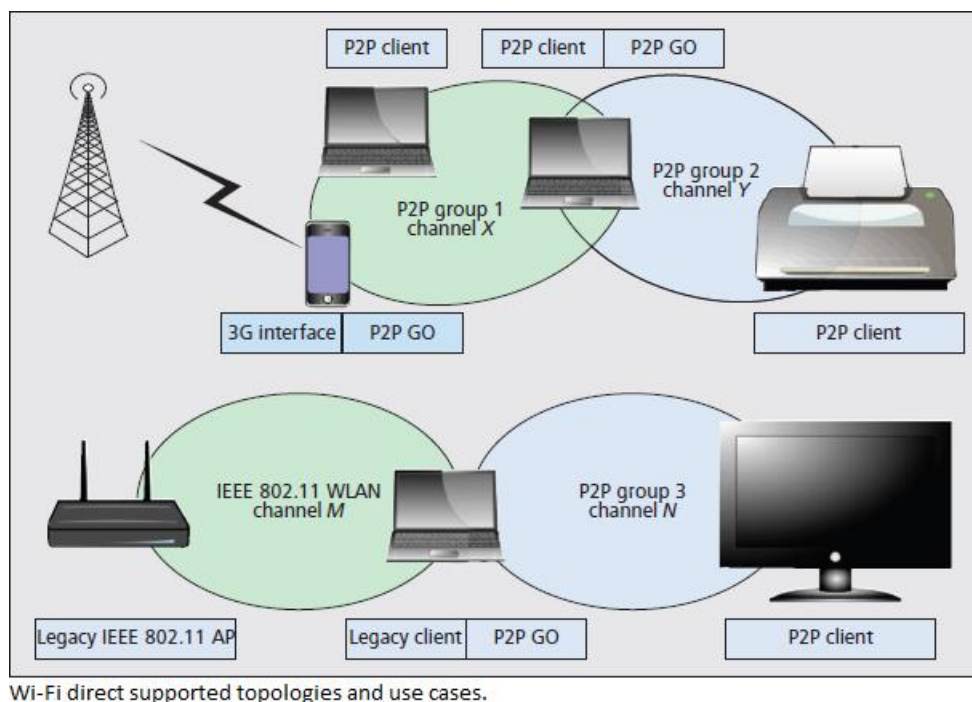
<sup>2</sup> wireless access point

<sup>3</sup> client

<sup>4</sup> dynamic

<sup>5</sup> Time sharing





شکل ۲-۱: معماری Wifi Direct [۱]

مانند یک نقطه‌ی دسترسی بی سیم سنتی، یک P2P Group Owner، خود را از طریق beacons اعلام می‌کند. تنها دستگاهی که P2P Group Owner است؛ قادر است دستگاه‌های متصل در گروه خود را به یک شبکه‌ی خارجی متصل کند. (مثال: موبایل موجود در بالای شکل ۲-۱) این ارتباط باید در لایه‌ی شبکه<sup>۶</sup> اتفاق بیفتد و معمولاً با استفاده از NAT<sup>۷</sup> پیاده سازی می‌شود. WiFi Direct اجازه نمی‌دهد که نقش P2P Group Owner به افراد دیگری در گروه انتقال یابد.

## ۲-۴ تشکیل گروه

سه نوع روش برای تشکیل گروه در فناوری WiFi Direct وجود دارد که عبارتند از استاندارد<sup>۸</sup>، مستقل<sup>۹</sup> و پایدار<sup>۱۰</sup>.

<sup>۶</sup>Network Layer

<sup>۷</sup>Network Address Translation

<sup>۸</sup>Standard

<sup>۹</sup>Autonomous

<sup>۱۰</sup>Persistent

تشکیل گروه شامل دو مرحله است:

#### ۱. تعیین P2P Group Owner

- دو دستگاه با توجه به تمایل یا قابلیت برای P2P Group Owner شدن با یکدیگر مذاکره می کنند.
- در نهایت در این مرحله نقش مالک گروه در سطح اپلیکیشن ایجاد می شود.

#### ۲. تهیه ی P2P Group

- ایجاد session گروه با استفاده از مدارک<sup>۱۱</sup> معتبر
- استفاده از پیکربندی<sup>۱۲</sup> ساده Wifi برای تبادل مدارک.

#### ۲-۴-۱ استاندارد

در این حالت دستگاه‌ها باید یکدیگر را پیدا<sup>۱۳</sup> کنند و سپس مذاکره کنند که کدام دستگاه به عنوان مالک گروه عمل خواهد کرد. شروع آن با انجام یک اسکن مانند WiFi سنتی است که با استفاده از آن می توانند گروه‌ها و شبکه‌های WiFi موجود را پیدا کنند. برای جلوگیری از تضاد، زمانی که دو دستگاه آمادگی خود را برای مالک گروه شدن اعلام می کنند، یک بیت tie-breaker در درخواست قرار می گیرد. هر بار که یک درخواست ارسال می شود، به طور تصادفی این بیت تنظیم می شود. [۱]

#### ۲-۴-۲ مستقل

یک دستگاه می تواند به صورت خودکار یک گروه ایجاد کند و بلافاصله این دستگاه مالک گروه می شود. دستگاه‌های دیگر می توانند گروه‌های ایجاد شده را با استفاده از مکانیزم‌های سنتی اسکن پیدا کنند. در مقایسه با بخش ۲-۴-۱، مرحله پیدا کردن در این مورد ساده تر است، زیرا مرحله مذاکره برای مالک گروه شدن حذف شده است. [۱]

<sup>11</sup>Credentials

<sup>12</sup>Configuartion

<sup>13</sup>Discovery

## ۲-۴-۳ پایدار

در این فرآیند، دستگاه می‌تواند با استفاده از پرچم<sup>۱۴</sup> که به عنوان یک ویژگی در فریم‌های beacon موجود است یک گروه را به عنوان گروه پایدار اعلام کند. پس از مرحله پیدا کردن، اگر یک دستگاه تشخیص دهد که یک گروه پایدار با همتای مربوطه در گذشته تشکیل داده باشد، هر یک از دو دستگاه می‌تواند از روش دعوت<sup>۱۵</sup> برای استفاده سریع از گروه مجدداً استفاده کنند. [۱]

## ۲-۵ امنیت

دستگاه‌های Wifi Direct از WPS<sup>۱۶</sup> پشتیبانی می‌کنند. WPS یک اتصال امن را با معرفی یک PIN در مشتری یا فشار دادن یک دکمه در دو دستگاه P2P ایجاد می‌کند. [۱]

## ۲-۶ ذخیره انرژی

Wifi Direct دو مکانیسم جدید ذخیره انرژی را به کار می‌گیرد:

۱. پروتکل Opportunistic Power Save

۲. پروتکل Notice of Absence

## ۲-۶-۱ Opportunistic Power Save protocol

این پروتکل این اجازه را به مالک گروه می‌دهد که زمانی که تمامی مشتری‌های گروه در حالت خواب<sup>۱۷</sup> هستند؛ انرژی خود را ذخیره کند.

<sup>14</sup>Flag

<sup>15</sup>Invitation

<sup>16</sup>WiFi Protected Setup

<sup>17</sup>Sleep

## ۲-۶-۲ Notice of Absence (NOA) protocol

این پروتکل این اجازه را به مالک گروه می‌دهد که فواصل زمانی را که به آن‌ها دوره‌های زمانی غیابی می‌گویند را اعلام کند که در این دوره‌های زمانی، مشتریان مجاز به دسترسی به کانال نیستند. مالک گروه یک برنامه NOA را با استفاده از چهار پارامتر زیر تعریف می‌کند:

- مدت زمانی که طول هر دوره غیبت را مشخص می‌کند
- فاصله زمانی که بین دوره‌های غیبت متوالی وجود دارد
- زمان شروع اولین دوره غیبت پس از فریم beacon کنونی
- تعداد دوره‌های غیبت برنامه ریزی شده [۱]

## ۲-۷ فواید

در این بخش به فواید و مزایای Wifi Direct می‌پردازیم.

۱. تحرک و قابلیت حمل: دستگاه‌هایی که قابلیت Wifi Direct را دارند در هر مکانی و در هر زمانی می‌توانند به یکدیگر متصل شوند.
۲. سهولت استفاده: دستگاه‌های دارای Wifi Direct ویژگی‌هایی را دارند که کاربران را قادر می‌سازد تا قبل از برقراری ارتباط، دستگاه‌ها و خدمات موجود را شناسایی کنند.
۳. اتصال ساده امن: Wi-Fi Protected setup باعث ساده ساختن ارتباطات محافظت شده بین دستگاه‌ها می‌شود. کاربران در بیشتر موارد قادر به اتصال با یک دکمه خواهند بود. [۷]

## فصل ۳

# طراحی و پیاده‌سازی

### ۳-۱ اتصال دستگاه‌ها در لایه‌ی فیزیکی با استفاده از فناوری WifiDirect

برای اجرای خدمات لایه‌ی شبکه‌ی اجتماعی، ابتدا لازم است که دستگاه‌ها در لایه‌ی فیزیکی به یکدیگر متصل شوند و امکان ارسال و دریافت اطلاعات در لایه‌ی فیزیکی فراهم شود. به همین جهت از فناوری WiFi Direct استفاده خواهیم کرد تا شرایط لازم در لایه‌ی فیزیکی را برای ما برقرار کنند. در ادامه‌ی این بخش توضیح خواهیم که سه مورد Discovery ، Connect و Create Group برای برقراری ارتباط در لایه‌ی فیزیکی چگونه در WiFi Direct انجام می‌شود.

### ۳-۱-۱ Discovery

برای برقراری ارتباط، باید بررسی کنیم که چه افرادی در نزدیکی ما حضور دارند. به این کار Discovery می‌گوییم. برای انجام Discovery در WiFi Direct کافی است از متد discoverPeers() استفاده کنیم. با صدا زدن این متد، Discovery آغاز می‌شود و در صورت موفقیت‌آمیز بودن، با صدا زدن متد requestPeers() می‌توانیم به لیست افرادی که در نزدیکی ما هستند؛ دست پیدا کنیم و تمامی افرادی که آمادگی لازم برای برقراری ارتباط دارند را مشاهده کنیم. توجه داشته باشید که در این روش تنها افرادی را می‌توان مشاهده کرد که آن‌ها هم عملیات Discovery را انجام دهند. در غیر این صورت شما نمی‌توانید آن‌ها را در لیست خود مشاهده کنید.

### ۳-۱-۲ Connect

بعد از پیدا کردن افراد نزدیک به خودمان، باید بتوانیم به فرد مورد نظر متصل شویم. برای این کار با دانستن اطلاعات پیکربندی دستگاه مورد نظر و با استفاده از متد `connect()` عملیات اتصال را آغاز می‌کنیم. اگر متد `connect()` موفقیت آمیز باشد، با استفاده از رابط `ConnectionInfoListener` بررسی می‌کنیم که آیا وضعیت اتصال دستگاه فعلی تغییر کرده است یا خیر. با استفاده از متد `onConnectionInfoAvailable()` که رابط `ConnectionInfoListener` در اختیار ما قرار می‌دهد، چک می‌کنیم که کدام یک از این دستگاه‌ها مالک گروه هست و کدام یک مشتری گروه هست. بنابراین با توجه به نقش هر دستگاه در این ارتباط نقش `Server` یا `Client` را برای آن‌ها انتخاب خواهیم کرد. توجه شود که اطلاعات پیکربندی دستگاه شامل مواردی مانند آدرس <sup>۱</sup> MAC دستگاه و اطلاعات WPS <sup>۲</sup> می‌باشد.

### ۳-۱-۳ Create Group

در بخش قبل توضیح دادیم که چگونه می‌توانیم دو دستگاه را به یکدیگر متصل کنیم. اما اگر بخواهیم چندین دستگاه به یکدیگر متصل شوند؛ باید از متد `createGroup()` استفاده کنیم. در این صورت دستگاهی که این متد را صدا بزند به عنوان مالک گروه خواهد شد و هر دستگاهی که وارد این گروه شود به عنوان مشتری گروه خواهد بود. پس از این که این متد به صورت موفقیت‌آمیز اجرا شد؛ با فراخوانی متد `requestGroupInfo()` می‌توانیم جزئیات بیشتری در مورد دستگاهی‌هایی که در این گروه قرار دارند مانند نام این دستگاه‌ها و وضعیت اتصال آن‌ها را درخواست بدهیم. سپس با استفاده از متد `onGroupInfoAvailable()` که رابط `GroupInfoListener` در اختیار ما قرار می‌دهد؛ می‌توانیم اطلاعات مرتبط با گروه را مشاهده کنیم.

## ۲-۳ دریافت و ارسال پیام

در بخش قبل دیدیم که چگونه می‌توان با استفاده از `WiFi Direct` دستگاه‌ها را در لایه فیزیکی به یکدیگر متصل کرد. در این بخش می‌خواهیم با استفاده از برنامه‌نویسی `socket` و از طریق دو روش `Multi thread` و `Asynchronous` این امکان را ایجاد کنیم که دستگاه‌ها با یکدیگر پیام رد و بدل کنند. بنابراین در یک ارتباط،

<sup>۱</sup>Media Access Control

<sup>۲</sup>Wi-Fi Protected Setup

لازم است که یکی از دستگاه‌ها در نقش server و دیگری در نقش client باشد. ما در اینجا دستگاهی که به عنوان مالک گروه باشد را server و دیگری را client در نظر می‌گیریم. در ادامه‌ی این بخش وظیفه‌ی server و client را در هر دو روش Multi thread و Asynchronous بیان می‌کنیم و توضیح خواهیم داد که هر کدام از چه متدهایی استفاده خواهند کرد.

### ۳-۲-۱ Multi thread یا Synchronous

همانطور که از اسم این روش پیداست، در این روش، برای هر قسمت از کاری که می‌خواهیم انجام بدهیم برای مثال اتصال، ارسال پیام و دریافت پیام و ... یک thread اختصاص می‌دهیم که همه‌ی آن‌ها به صورت همزمان در حال انجام وظیفه‌ی خودشان هستند. در این روش ما منتظر جواب از طرف مقابل هستیم و در واقع تا دریافت جواب، آن متد مسدود یا Block می‌شود. شبیه زمانی است که شما در حال مکالمه با دوستان در پشت تلفن هستید. وقتی که شما حرفی می‌زنید، منتظر هستید تا نفر دیگر پاسخ شما را بدهد. بنابراین همین انتظار باعث می‌شود تا زمان زیادی مصرف شود و منابع ما هدر برود.

همچنین این روش به صورت دنباله‌ای متوالی انجام می‌شود. یعنی شما یک کانال برای ارتباط ایجاد می‌کنید (شماره‌ی دوستان را می‌گیرید) و در ادامه یک سری اطلاعات را تبادل می‌کنید (شما یا دوستان صحبت می‌کنید و منتظر جواب طرف مقابل هستید). و در آخر هم کانال ارتباطی را می‌بندید (تلفن را قطع می‌کنید). در این روش منابع موجود، بین thread ها به اشتراک گذاشته می‌شود. این روش برای تعداد اتصالات کم مناسب است. اما وقتی تعداد اتصالات وسیع باشد؛ دیگر نمی‌توان همه‌ی اتصالات را به صورت درست مدیریت کرد. در واقع برنامه‌ریزی <sup>۳</sup> thread ها سربار ایجاد خواهد کرد. زیرا تعدادشان زمانی که اتصالات زیاد می‌شود؛ افزایش پیدا می‌کند.

برای پیاده‌سازی این روش از سه کلاس استفاده کرده‌ایم که در ادامه هر کدام را به صورت مجزا توضیح خواهیم داد.

Server

در کلاس Server دو متد را پیاده‌سازی کردیم:

<sup>3</sup>Scheduling

۱. `Accept()` : در این متد `server` در یک حلقه‌ی بی‌نهایت بر روی درگاه ۸۸۸۸ در حال گوش دادن به درخواست‌های `client` است. بعد از آن که `server` درخواست یک `client` را قبول می‌کند، برای این `client` یک `ClientHandler` می‌سازد و آن را به لیست `Client` هایی که به سرور متصل هستند، اضافه می‌کند. متد `Accept()` برای انجام همه‌ی این مراحل از یک `thread` استفاده می‌کند.

۲. `Send()` : در این متد `server` یک پیام را برای همه‌ی `client` هایی که به آن متصل هستند؛ ارسال می‌کند. در واقع پیام را بر روی جریان خروجی داده که توسط `socket` ما بین `server` و `client` ایجاد شده است؛ قرار می‌دهد. برای انجام این کار، از یک `thread` استفاده می‌شود.

### Client

در کلاس `Client` سه متد را پیاده‌سازی کردیم:

۱. `Connect()` : در این متد `client` به `server` ای که در حال گوش دادن بر روی درگاه ۸۸۸۸ هست؛ متصل می‌شود. به محض این که `socket` به درستی بین `client` و `server` ایجاد شد، `client` با صدا زدن متد `Receive()` آماده‌ی دریافت پیام از سوی `server` می‌شود. توجه شود که همه‌ی این کارها توسط یک `thread` انجام می‌شود.

۲. `Send()` : در این متد `Client` پیامی را که می‌خواهد برای `server` ارسال کند را بر روی جریان خروجی داده که به کمک `socket` بین این دو ایجاد شده است، قرار می‌دهد. از یک `thread` برای ارسال پیام استفاده می‌کنیم.

۳. `Receive()` : در این متد پیامی را که `server` برای `client` ارسال کرده است را از روی جریان ورودی داده می‌خوانیم. همانند ارسال برای دریافت هم از یک `thread` استفاده می‌کنیم.

### ClientHandler

هدف ما در این کلاس این است که بتوانیم `client` های متصل به `server` را مدیریت کنیم. این کلاس تنها یک متد `Broadcast()` را دارد. در این متد قرار است که `server` پیام‌هایی را که از `client` هایش دریافت می‌کند را برای سایر `client` هایش ارسال کند. کاری که دقیقاً در این متد انجام می‌شود؛ این است که پیام را از جریان



ورودی داده‌ی یکی از client هایش می‌خواند و سپس آن را بر روی جریان خروجی داده سایر client هایش قرار می‌دهد. دقت شود که همه‌ی این کارها توسط یک thread خاص اجرا می‌شود.

### ۳-۲-۲ Asynchronous

در این روش ما کارهای لازم در شبکه را پست می‌کنیم ولی منتظر جواب نمی‌مانیم بلکه نتیجه را بعدا بررسی می‌کنیم. شبیه موقعی است که شما به دوستان پیام متنی ارسال می‌کنید و منتظر جواب در همان لحظه نمی‌مانید و جواب دوستان ممکن است الان یا یک ساعت دیگر برسد. بنابراین جواب را بعدا چک می‌کنید.

این روش بر خلاف روش synchronous به صورت دنباله‌ای نیست. اما دنبال کردن وضعیت هر یک از کارها کمی مشکل است. بنابراین با توجه به ویژگی‌هایی که برای این روش ذکر کردیم؛ می‌توانیم با استفاده از این روش از هدر رفتن منابع جلوگیری کنیم. همچنین می‌توانیم تعداد اتصالات بیشتری را با همان سخت افزار موجود مدیریت کنیم.

برای پیاده‌سازی این روش از ۴ کلاس استفاده کرده‌ایم. که در ادامه هر کدام را به صورت مجزا توضیح خواهیم داد.

#### WiFiNetService

هدف از این کلاس این است که دستگاه چه در نقش client و چه در نقش server باشد، بتواند ۴ کار ارتباطی زیر را انجام دهد:

##### ۱. ارسال پیام به یک دستگاه خاص:

برای انجام این کار از متد Send() استفاده می‌کنیم که در آن از متد StartWrite() استفاده شده است که بوسیله‌ی آن می‌توان پیام را بر روی بافر دستگاه مورد نظر قرار داد. چون نوشتن بر روی بافر، یک عمل ناهمزمان است از CompletionHandler استفاده می‌کنیم که کامل شدن یا شکست خوردن را به thread اصلی اعلام کند.

##### ۲. دریافت پیام از یک دستگاه خاص:

برای انجام این کار از متد Receive() استفاده می‌کنیم که در آن از متد StartRead() استفاده شده است که بوسیله‌ی آن می‌توان پیام را از بافر دستگاه مورد نظر خواند. چون خواندن از روی بافر یک عمل

ناهمزمان است از `CompletionHandler` استفاده می‌کنیم که کامل شدن یا شکست خوردن را به `thread` اصلی اعلام می‌کند.

۳. ارسال پیام به تمامی دستگاه‌هایی که به دستگاه فعلی متصل هستند:

برای انجام این کار از متد `Broadcast()` استفاده می‌کنیم که در آن پیام بر روی بافر همه‌ی دستگاه‌هایی که به دستگاه فعلی متصل هستند؛ قرارداد می‌شود. این عمل نیز به صورت ناهمزمان اجرا می‌شود.

۴. ارسال پیام دریافت شده به تمامی دستگاه‌هایی که به دستگاه فعلی متصل هستند:

برای انجام این کار از متد `ReceiveBroadcast()` استفاده می‌کنیم که پس از دریافت یک پیام آن را بر روی بافر تمامی دستگاه‌های متصل به دستگاه فعلی قرار می‌دهیم. این عمل هم به صورت ناهمزمان اجرا می‌شود.

#### Device

هدف از ایجاد این کلاس این است که برای هر دستگاه یک کانال ناهمزمان `socket` برای رد و بدل پیام در نظر بگیریم. و همچنین اطلاعات دیگری نظیر اسم دستگاه را نگهداری کنیم.

#### Server

این کلاس تنها یک متد `Start()` دارد که با صدا زدن آن `server` در یک حلقه‌ی بی‌نهایت بر روی درگاه ۸۸۸۸ شروع به گوش دادن به درخواست از طرف `client` ها می‌کند. اما این کار باعث انسداد یا `Blocking` نمی‌شود. بلکه به صورت ناهمزمان انجام می‌شود. بنابراین از یک `CompletionHandler` استفاده کرده‌ایم. که در صورتی که یک `client` به `server` متصل شود و عمل کامل شود؛ سرور متد `ReceiveBroadcast()` را از کلاس `WiFiNetService` فراخوانی می‌کند و آماده‌ی دریافت پیام از یکی از `client` هایش می‌شود و سپس آن را برای تمامی دستگاه‌های متصل، ارسال می‌کند.

#### Client

این کلاس همانند کلاس `Server` تنها یک متد `Start()` دارد که با فراخوانی آن دستگاه به `server` ای که بر روی درگاه ۸۸۸۸ در حال گوش دادن هست، متصل می‌شود. این عمل نیز به صورت ناهمزمان انجام می‌شود.

بنابراین از یک `CompletionHandler` استفاده کرده‌ایم و در صورت کامل شدن اتصال، دستگاه با فراخوانی متد `Receive()` از کلاس `WiFiNetService` آماده‌ی دریافت پیام می‌شود.

## مراجع

- [1] CAMPS-MUR, D., GARCIA-SAAVEDRA, A., AND SERRANO, P. Device-to-device communications with wi-fi direct: overview and experimentation. *IEEE Wireless Communications Magazine* 20 (06 2013).
- [2] FOUNDATION, O. C. Open connectivity foundation. <https://openconnectivity.org/>, 2019.
- [3] HYPELABS. Sdk for connectivity even without internet. <https://hypelabs.io/>, 2018.
- [4] NEARBY GROUP. Nearby. <https://www.wnmlive.com/>, 2018.
- [5] OPENGARDEN. Introducing a new wireless carrier. <https://www.opengarden.com/>, 2018.
- [6] P2PKIT GROUP. p2pkit api. <http://p2pkit.io/>, 2016.
- [7] WIFL.ORG. Wi-fi certified wi-fi direct®: Personal, portable wi-fi® technology, October 2014.
- [8] WIKIPEDIA. Firechat. <https://en.wikipedia.org/wiki/FireChat>, 2019.
- [9] WIKIPEDIA. Nextdoor. <https://en.wikipedia.org/wiki/Nextdoor>, 2019.
- [10] WIKIPEDIA. Open garden. [https://en.wikipedia.org/wiki/Open\\_Garden](https://en.wikipedia.org/wiki/Open_Garden), 2019.

**Abstract:**

The goal of this project is to add a social network service layer to a wireless mesh network. In this project, a mobile application for the neighborhood service is first implemented. Then the necessary services from the social service layer are identified for this service. Then, the services of this layer are engineered, designed and implemented to provide wireless mesh network connectivity. Ultimately, with this app, we can launch a social network that can even provide online social services, such as sending private messages or public ones, even without an internet connection.

**Keywords:** Social Network, Wireless, Mesh Network, Mobile Application, Neighborhood



**Iran University of Science and Technology  
Computer Engineering Department**

# **Design and implementation of local and wireless social network without Internet**

**A Thesis Submitted in Partial Fulfillment of the Requirement for the Degree  
of Master of Science in Computer Engineering**

**By:**

**Maryam Sadat Hashemi**

**Supervisor:**

**Sayyed Sauleh Eetemadi**

**November 2019**