

دانشگاه صنعتي امیرکبیر  
(پلی تکنیک تهران)



دانشکده مهندسی کامپیوتر

گزارش کارآموزی

محل کارآموزی: شرکت کوییزآو کینگز

|  |  |
| --- | --- |
| نام استاد کارآموزی: | مهندس جعفر عبدی |
| نام دانشجو: | محمدرضا اخگری زیری |
| شماره دانشجویی: | 9631001 |

تابستان 1399

**تقدیر و تشکر**

بدینوسیله مراتب قدردانی و امتنان خود را خدمت،

جناب آقای امیرطاعی، سرپرست تیم توسعه دهنده سمت مشتری شرکت کوییز آوکینگز، بابت آموزش‎ها و زحمات بی‌دریغشان،

جناب آقای علی نظری، دبیر انجمن علمی دانشکده مهندسی کامپیوتر، بابت معرفی این دوره کارآموزی

و تمامی کارکنان و همراهان بنده در شرکت کوییزآوکینگز،

ابراز و از تمامی زحمات آنان تشکر می‌نمایم.

محمدرضا اخگری زیری

شهریور 99

چكيده

امروزه یکی از رسانه‎هایی که میزان قابل توجهی از افراد جامعه را متوجه خود ساخته، بازی‌های رایانه‌ای است. از نیمة دوّم قرن بیستم، همراه و همگام با تحوّلات و پیشرفت‌ها در صنایع الکترونیکی، بازی‌ها و اسباب بازی‌هایی جدید و متفاوت با گذشته، پا به عرصه وجود گذاشته که اصطلاحاً «بازی‌های رایانه‌ای» نامیده می‌شوند. صنعت ساخت بازی‌های ویدیویی و رایانه‌ای، در حال حاضر، به گسترده‌ترین و سودآورترین حرفه‌ در صنعت سرگرمی تبدیل شده‌است.

در این گزارش، پس از نیازسنجی شرکت کوییزآوکینگز، متوجه شدیم که موتور بازی‌سازی قبلی به نسخه متن‌باز[[1]](#footnote-1) تبدیل شده و دیگر از نسخه 10 سیستم‌عامل آندروید[[2]](#footnote-2) پشتیبانی نمی‌کند. بنابراین مسائل، ما اقدام به تغییر پلتفرم بازی از موتور بازی‌سازی کرونا[[3]](#footnote-3)، به موتور یونیتی[[4]](#footnote-4) کردیم.

واژه‌های کلیدی:

بازی‌های رایانه‌ای، موتور بازی‌سازی، موتور کرونا، موتور یونیتی.

|  |  |
| --- | --- |
| فهرست مطالب | صفحه |

[فصل اول مقدمه 1](#_Toc51933743)

[1-1- مقدمه 2](#_Toc51933744)

[1-2- برنامه نویسی سمت مشتری و سرویس‌دهنده 3](#_Toc51933745)

[1-2-1- سرویس‌‌دهنده 3](#_Toc51933746)

[1-2-2- وظایف سرویس‌‎‎دهنده‌ها 4](#_Toc51933747)

[1-2-3- مشتری 4](#_Toc51933748)

[1-2-4- وظایف مشتری‎ها 4](#_Toc51933749)

[1-3- نتیجه‌گیری 5](#_Toc51933750)

[فصل دوم معرفی محل کارآموزی 6](#_Toc51933751)

[2-1- تاریخچه 7](#_Toc51933752)

[2-2- معرفی کلی شرکت 8](#_Toc51933753)

[2-2-1- بازی انفرادی 8](#_Toc51933754)

[2-2-2- لیگ ستاره 8](#_Toc51933755)

[2-2-3- لیگ دوستانه 8](#_Toc51933756)

[2-2-4- بازی گروهی 8](#_Toc51933757)

[2-2-5- رکورد 8](#_Toc51933758)

[2-3- افتخارات 8](#_Toc51933759)

[2-4- هدف 8](#_Toc51933760)

[2-5- جمع‌بندی 9](#_Toc51933761)

[فصل سوم فعالیت‌ها و تجارب کارآموزی 10](#_Toc51933762)

[3-1- انتخاب پلتفرم 11](#_Toc51933763)

[3-1-1- موتور کرونا 11](#_Toc51933764)

[3-1-1-1- فواید 11](#_Toc51933765)

[3-1-1-2- معایب 12](#_Toc51933766)

[3-1-2- موتور یونیتی 13](#_Toc51933767)

[3-2- تسلط بر کد 14](#_Toc51933768)

[3-2- مستند سازی 15](#_Toc51933769)

[3-2-1- API 15](#_Toc51933770)

[3-2-2- REST 16](#_Toc51933771)

[3-2-3- RESTful API 16](#_Toc51933772)

[3-3- زبان YAML 17](#_Toc51933773)

[3-4- یادگیری یونیتی 18](#_Toc51933774)

[3-5- طراحی 19](#_Toc51933775)

[3-5-1- معماری MVC 19](#_Toc51933776)

[3-5-2- Model 20](#_Toc51933777)

[3-5-3- View 20](#_Toc51933778)

[3-5-4- Controller 21](#_Toc51933779)

[3-6- آکادمی 21](#_Toc51933780)

[3-7- متدولوژی اسکرام 21](#_Toc51933781)

[فصل چهارم نتیجه گیری 23](#_Toc51933782)

[4-1- جمع بندی 24](#_Toc51933783)

[4-2- کارهای آینده 24](#_Toc51933784)

[منابع و مراجع 25](#_Toc51933785)

|  |  |
| --- | --- |
| فهرست شکل‌ها | صفحه |

[شکل 1-2- نمایی از نسخه‎ی اولیه بازی کوییزآو کینگز 7](#_Toc51933786)

[شکل 2-2- نمایی از نسخه فعلی بازی کوییزآو کینگز 8](#_Toc51933787)

[شکل 1-3- نمای یک API 16](#_Toc51933788)

[شکل 2-3- معماری یک REST API 17](#_Toc51933789)

[شکل 3-3- نمایش معماری MVC 20](#_Toc51933790)

# فصل اول مقدمه

## 1-1- مقدمه

به باور روان‌شناسان و جامعه‌شناسان، بازی که عمیقاً با زندگی پیوند خورده است، در روحیه و تکوین شخصیت افراد، اثر عمیقی بر جای می‌گذارد و باعث می‌شود که فرد از طریق آشنایی با الگوها و هنجارهای جامعة خویش و درونی‌سازی ارزش‌های آن در راستای اجتماعی شدن، گام بردارد[1].

پدیده‌ی بازی بر حسب علل، زمینه‌ها و شرایط، طی زمان‌ها و مکان‌های مختلف، به لحاظ ساختار، محتوا و کارکرد (اثر)، شکل‌های گوناگونی به خود گرفته است.

اگرچه تاریخچه‌ی بازی‌های رایانه‌ای به دهه‎ی 1970م، برمی‌گردد؛ اما در مدت کوتاهی پس از پیدایش، نه تنها تحول عمیق و اساسی در نظام تفریحات و سرگرمی کودکان به‌وجود آوردند، بلکه در حال حاضر، گستره‌ی بازی‌های رایانه‌ای به عنوان یک وسیله‌ی گذران اوقات فراغت، به عرصه‌ی بزرگسالان نیز کشیده شده است.

بازی‌های الکترونیکی در عین آن که خود دارای شبکه، سایت‌های مخصوص، گروه‌های گفت وگو و صحبت[[5]](#footnote-5) در اینترنت هستند، به صورت مجموعه‌ای صوتی و تصویری (برنامه‌های ویدیویی و تلویزیونی) به دیگر رسانه‌ها منتقل شده‌اند.

در این بین، رقابتی که بین شرکت‌ها و تولیدکنندگان مختلف در تصاحب هرچه بیشتر بازار فروش به وجود آمده، آن‌ها را بر آن داشته که با ارائه‌ی بازی‌های هرچه جالب‌تر و مهیج‌تر، سود بیشتری را به خود اختصاص دهند.

شرکت دانش‌پروران امیرکبیر نیز به عنوان یکی از فعالان این زمینه، بازی‌ تحت شبکه‎ای به نام کوییزآو کینگز[[6]](#footnote-6) ارائه داده است. هدف این بازی علاوه بر سرگرمی، افزایش سطح علمی بازیکنان است.

## 1-2- برنامه نویسی سمت مشتری و سرویس‌دهنده[2]

در هر برنامه تحت شبکه‎ای، ما مفاهیمی به نام مشتری[[7]](#footnote-7) و سرویس‌دهنده[[8]](#footnote-8) را شنیده‌ایم، لازم است تا در ابتدای کار توضیحی از این مفاهیم ارائه دهیم تا در ادامه بتوانیم درباره‎ی آن‌ها صحبت کنیم.

واژه‌های سرویس‌دهنده و مشتری، در دهه‌ی 80 میلادی همزمان با عرضه‌ی تجهیزات شبکه به بازار، مطرح شد. تصور افراد از مشتری و سرویس‌دهنده صرفاً به تجهیزات سخت‌افزاری کامپیوترها مربوط می‌شد. درآن دوران، سرویس‌دهنده به کامپیوتری اطلاق می‌گردید که در حقیقت مانند مرکز ثقل ارائه خدمات در شبکه بود و مشتری نیز کامپیوتری بود که این امکانات را مورد استفاده قرار می‌داد. این تعاریف اگرچه درست بودند اما ناقص بودند. یعنی این تعریف که مشتری، صرفاً یک کامپیوتر فیزیکی است و سرویس‌دهنده، کامپیوتر فیزیکی متصل به آن است که وظیفه ارائه سرویس را برعهده دارد، تعریف کاملی نیست. در ادامه به بررسی این دو مفهوم به طور کامل می‌پردازیم.

### 1-2-1- سرویس‌‌دهنده

سرویس‌دهنده، یک کامپیوتر همواره روشن است که در تمامی مواقع در دسترس است و به کامپیوترها و تجهیزات دیجیتالی که به شبکه متصل هستند، سرویس می‌دهد. و به همین دلیل به آن، سرویس‌دهنده می‌گویند. اساساً راه‌‌اندازی سرویس‌دهنده‌ها برای ارائه خدمت به تجهیزات متصل در شبکه انجام می‌شود تا بتوانند به درخواست‌های مشتریان پاسخ دهند. به عنوان نمونه، در زمان اتصال شما به یک وب‌سایت و استفاده از خدمات آن، سرویس‌دهنده آن سایت در حال خدمت‌رسانی به شما است. سرویس‌دهنده‌ها خدمات متنوعی در اختیار مشتری قرار می‌دهند که از آن جمله می‌توان به انتقال فایل[[9]](#footnote-9)، سرویس وب، سرویس بازی[[10]](#footnote-10)و دیگر سرویس‌دهنده‌ها اشاره کرد.

### 1-2-2- وظایف سرویس‌‎‎دهنده‌ها

از آنجایی که سرویس‎دهنده‌ها وظیفه ارائه خدمت به مشتری‎ها را دارند، بدیهی است که به لحاظ سخت‌افزاری، کامپیوترهای قدرتمندی محسوب می‌شوند. اشتراک منابع برای استفاده مشتری‎ها در شبکه توسط سرویس‎دهنده صورت می‌گیرد. اختصاص حافظه، حافظه اصلی، پردازشگر به مشتری‎های شبکه، میزان استفاده از اینترنت از جمله منابعی است که توسط سرویس‌دهنده به سرویس‌گیرندگان در شبکه اختصاص می‌یابد. سرویس‎دهنده‎ها، تعیین‌کننده میزان استفاده از منابع شبکه توسط مشتری‎ها هستند.

یک سرویس‌دهنده می‌بایست به درخواست مشتری‎ها و هر سرویس‌گیرنده‌ای در شبکه پاسخ دهد. زمانی که دو سرویس‌گیرنده که کاربر محسوب می‌شوند قصد تبادل پیغام و اطلاعات دارند، سرویس‌دهنده ابتدا داده را ذخیره و سپس به سرویس‌گیرنده دیگر ارسال می‌کند.

### 1-2-3- مشتری

در مقابل سرویس‎دهنده، تجهیزات و کامپیوترهایی که از سرویس‎دهنده، خدمات دریافت می‌کنند مشتری نامیده می‌شوند. تمامی تجهیزات، کامپیوترها و یا نرم‌افزارهای متصل به سرویس‌دهنده که که سرویس‌دهنده به آن‌ها خدمت ارائه می‌دهد، سرویس‌گیرنده محسوب می‌شوند. برای مثال، کامپیوترهای موجود در شبکه که به چاپگر متصلند، از آنجایی که از چاپگر شبکه، درخواست خدمات چاپ اطلاعات دارند، سرویس‌گیرنده شبکه محسوب می‌شوند.

### 1-2-4- وظایف مشتری‎ها

وظیفه اصلی مشتری‎ها، ارسال درخواست‌های خود به سرویس‌دهنده است. باتوجه به نوع نیاز هر مشتری،‌ درخواست‌ها توسط آن از طریق برنامه‌های کاربردی به سرویس‎دهنده ارسال می‌گردد. می‌توانیم درنظر بگیریم که مشتری از سرویس‎دهنده تقاضای کار[[11]](#footnote-11) می‌نماید.

ارسال و به اشتراگ‌گذاشتن اطلاعات در شبکه از دیگر وظایف مهم مشتری‎ها محسوب می‌گردد. نمونه آن زمانی است که کاربران در یک سرویس‎دهنده گفت‎وگو[[12]](#footnote-12)، اطلاعات خود را به سمت سرویس‎دهنده ارسال می‌نمایند و سپس ارسال اطلاعات از کاربر به کاربر دیگر از سوی سرویس‎دهنده انجام می‌شود.

## 1-3- نتیجه‌گیری

حال که به تفاوت این دو مفهوم در یک برنامه پی بردیم، لازم به ذکر است که ما در دوره‎ی کارآموزی، توسعه‎ی این برنامه در سمت مشتری را آغاز کردیم. اما به علت مشکلاتی که در ادامه ذکر خواهیم کرد، اقدام به تغییر پلتفرم و زبان برنامه‎نویسی سمت مشتری کردیم.

برای این اقدام، نیاز داشتیم تا به تسلط کافی از کد قبلی و محیط بازی برسیم. همچنین باید به تمام APIهای مورد نیاز برای بازنویسی برنامه آشنا بودیم. شرح این مسائل در فصل سوم به صورت کاملاً ملموس شرح داده شده است.

# فصل دوم معرفی محل کارآموزی

## 2-1- تاریخچه

در سال‎‌های 1394 الی 1396، جمعی از دانشجویان دانشگاه امیرکبیر، با ایده‎ای مبنی بر پیاده‎‌سازی یک بازی موبایل‌همراه با موضوع اطلاعات عمومی دور هم جمع شدند. این جمع، نسخه‌ی اولیه از بازی را ایجاد کردند و بر روی فروشگاه‎های برنامه‎ها از جمله بازار و گوگل استور[[13]](#footnote-13) قرار دادند.

بازی با استقبال بسیار خوبی از سمت کاربران مواجه شد و همین باعث شد تا شرکت دانش‌پروران امیرکبیر (کوییزآو کینگز) در سال 1396 با مالکیت خصوصی تأسیس شود. در شکل 1-2 نمایی از نسخه اولیه این بازی نمایش داده شده‎است.



شکل 1-2- نمایی از نسخه‎ی اولیه بازی کوییزآو کینگز

کوییز آو کینگز، یک بازی برخط[[14]](#footnote-14)، در سبک کلمات و دانستنی، ساخت شرکت دانش پروران امیرکبیر است. این بازی در لیگ بازی‌های رایانه‌ای ایران که توسط [بنیاد ملی بازی‌های رایانه ای](https://fa.wikipedia.org/wiki/%D8%A8%D9%86%DB%8C%D8%A7%D8%AF_%D9%85%D9%84%DB%8C_%D8%A8%D8%A7%D8%B2%DB%8C%E2%80%8C%D9%87%D8%A7%DB%8C_%D8%B1%D8%A7%DB%8C%D8%A7%D9%86%D9%87%E2%80%8C%D8%A7%DB%8C) در سال‌های ۱۳۹۶ و ۱۳۹۷ برگزار شد، پرمخاطب‌ترین بازی در مسابقات انتخاب شد و جایزه ویژه این مسابقات به نفرات برتر این بازی تعلق گرفت.



شکل 2-2- نمایی از نسخه فعلی بازی کوییزآو کینگز

## 2-2- معرفی کلی شرکت

کوییز آو کینگز، بزرگ‌ترین استارت‌آپ حوزه‌ی بازی‌های ویدئویی در کشور با بیش از ۶۰ نیروی فعال است. میانگین سنی این نیروهای حرفه‌ای، ۲۶ سال بوده و همواره سعی شده است تا کوییز آو کینگز روحیه‌ی جوان خود را حفظ نماید.

محور اصلی بازی کوییز آو کینگر بر روی رقابت برخط استوار است و رقابت در این بازی بر سر دانش و اطلاعات عمومی بازیکن‌ها صورت می‌پذیرد. این رقابت در پنج حالت انفرادی، لیگ ستاره، لیگ دوستانه، بازی گروهی و رکورد با ۱۹ موضوع مختلف قابل انجام است. این تغییرات به مرور زمان دربازی ایجاد شده که در شکل 2-2 تغییرات حاصله را مشاهده می‌‎کنید. بخش‌های مختلف بازی در ادامه شرح داده‎می‌شود:

### 2-2-1- بازی انفرادی

در هر مسابقه انفرادی کوییز آو کینگز دو بازیکن به عنوان رقیب یکدیگر باید به ۱۸ سؤال چهارگزینه‌ای پاسخ دهند. این سوالات در قالب ۶ موضوع مختلف پرسیده می‌شود و بازیکنان به صورت یک در میان موضوع سوالات را انتخاب می‌کنند.

### 2-2-2- لیگ ستاره

در هر رقابت لیگ ستاره، ده بازیکن باید به ۱۲ سؤال چهارگزینه‌ای در موضوعات مختلف پاسخ دهند و در پایان بازیکنان بر اساس تعداد پاسخ صحیح و زمان پاسخگویی، رتبه‌بندی می‌شوند.

### 2-2-3- لیگ دوستانه

لیگ دوستانه نیز مانند لیگ ستاره است ولی با این تفاوت که بازیکن، حریفان خود را از بین دوستان خود انتخاب می‎کند.

### 2-2-4- بازی گروهی

در بازی گروهی افراد یک گروه در قالب یک تیم به رقابت با گروهی دیگر می‌پردازند. هر گروه شامل تعدادی بازیکن است. تعداد سوالات در این حالت ۱۰ سؤال از تمامی موضوع‌های بازی است.

### 2-2-5- رکورد

در این بخش هر بازیکن به صورت شخصی به جواب دادن سوالات موضوعات مختلف به صورت متوالی پرداخته و فقط سه بار امکان پاسخ اشتباه وجود دارد. بعد از پایان بازی، رکورد او در جدول ثبت شده و برترین رکوردهای روزانه و هفتگی در جدول رده‌بندی مشخص می‌شوند.

## 2-3- افتخارات

شرکت کوییزآو کینگز موفق به کسب افتخارات زیر در طی سالیان گذشته شده است:

* برترین بازی سال‌های ۹۵ و ۹۶ (کلمات و دانستنی) از دید داوران و مردم.
* برترین بازی سال ۹۷ (کلمات و دانستنی) از دید مردم در جشنواره وب و موبایل ایران.
* بهترین بازی مردمی سال ۹۵ در ششمین جشنواره بازی‌های رایانه‌ای.
* بازی سال ۹۷ در بخش بهترین به‌روزرسانی از دیدمردم در هشتمین جشنواره بازی‌های رایانه‌ای.

## 2-4- هدف

محتوای بازی‎ها به سرعت در حال تغییر است و به دلیل کاربردهای گرافیکی و جاذبه ای که در آنها نهفته است، روز به روز واقعی تر به نظر می‌رسند و نظر افراد کثیری را به خود جلب می نمایند. متأسفانه موضوعی که در این آشفتگی بازار مورد توجه نیست، همان جنبه های اخلاقی و تربیتی کاربران این بازی ها است. از این رو شرکت کوییزآو کینگز با ایجاد این بازی سعی در بالا بردن معلومات عمومی بازیکنان در کنار سرگرمی دارد.

## 2-5- جمع‌بندی

ما در این شرکت در دوره کارآموزی وظیفه توسعه بخش مشتری این برنامه را به عهده داشتیم. دوره کارآموزی بنده به دلیل بیماری کرونا، به صورت دورکاری برگزار شد و محیط کار اینجانب درون منزل بود.

# فصل سوم فعالیت‌ها و تجارب کارآموزی

بنیانگذاران شرکت کوییزآو کینگز، برای پیاده‌سازی سریع نسخه ابتدایی خود از موتور بازی‌سازی کرونا که از زبان لوا پشتیبانی می‌کرد، استفاده کرده بودند. اما این موتور بازی‌سازی علی‌رغم فوایدی که داشت، مشکلاتی نیز به همراه داشت که تصمیم به تغییر پلتفرم گرفته شد، در طول کارآموزی اقدام به تغییر پلفترم کردیم.

## 3-1- انتخاب پلتفرم

موتور بازی سازی کرونا با مشکلاتی که به همراه داشت، باعث شد تا شرکت، تصمیم به تغییر پلتفرم بگیرد. اولین چالش ما، انتخاب موتور جایگزین بود. با بررسی موتورهای موجود و مقایسه آن‌ها، موتور یونیتی را انتخاب کردیم. برای درک بهتر این وضعیت، در ادامه به مقایسه این دو موتور بازی‌سازی میپردازیم، برای این منظور، ابتدا نیاز داریم تا فواید و معایب موتور بازی‌سازی کرونا را بیان کنیم و سپس علت انتخاب موتور یونیتی را به تفصیل شرح دهیم.

### 3-1-1- موتور کرونا[3]

توسعه بازی‌های تلفن‌همراه به طور خاص و انحصاری بسیار محبوب شده‌است. اکثر برنامه‌های کاربردی[[15]](#footnote-15) و بازی‌های جدید بر اساس موتورکرونا ساخته شده‌اند.

#### 3-1-1-1- فواید

در این بخش قصد داریم، با مزایای استفاده از موتور کرونا در بازی‌سازی آشنا شویم. این مزایا عبارتند از:

* **پشتیبانی از چندین پلتفرم[[16]](#footnote-16):** یکی از مزایای اصلی موتور کرونا این است که برنامه‌نویس، فقط کافیست یک‎ مرتبه کد خود را بنویسد و برای اجرای آن بر روی سیستم‌عامل‌های مختلف نیازی به تغییر و اصلاح کد نیست.
* **فریم بر ثانیه**:[[17]](#footnote-17) موتور کرونا به دلیل سرعت بالا و حداقل تاخیر در مقایسه با سایر SDK[[18]](#footnote-18)های موجود، در بازار امروز بسیار شناخته شده است. با استفاده از این SDK، شما می‌توانید به راحتی تمام ابعاد صفحه‌ای که نیاز دارید را تنظیم کنید. عموماً با استفاده از SDKهایی مانند PhoneGap و ImpactJS مقدار FPS بسیار کم است و توسعه بازی‌های تلفن‌همراه، یک تجربه عقب‌مانده محسوب می‌شود ولی با موتور کرونا این‌طور نیست.
* **شبیه‌ساز:** شبیه‌ساز استفاده شده در موتورکرونا کاملاً مشابه تلفن‌همراه شما است. در صورت استفاده از این SDK، شبیه‌ساز، پروژه شما را به عنوان یک فایل  ذخیره می‌کند و فرایند توسعه را بسیار سریع‌تر می‌سازد.
* **زبان برنامه‌نویسی لوا:** موتور کرونا از زبان برنامه‌نویسی لوا استفاده می‌کند و یکی از ویژگی‌های این زبان یادگیری آسان و استفاده راحت آن است.

#### 3-1-1-2- معایب

همه‎ی موتورهای بازی‌سازی در کنار مزایا، معایبی نیز دارند. این موتور نیز دارای معایبی می‌باشد که ادامه راه توسعه برنامه را با مشکل مواجه می‎کند. در ادامه به شرح این معایب می‌پردازیم:

* **قیمت:** رایج ترین مسئله‌ای که کاربران موتورکرونا با آن مواجه هستند، قیمت بالای آن است که باید برای استفاده از آن پرداخت شود.
* **عدم پشتیبانی از افزونه‌[[19]](#footnote-19)های غیربومی:** در صورتی که شما در پروژه خود نیاز به استفاده از یک افزونه دیگر داشته باشید، هیچ راهی برای ادغام افزونه‌های جدید نخواهید داشت. به این ترتیب ممکن است استفاده از موتور کرونا برای تمام برنامه های کاربردی شما امکان پذیر نباشد.
* **عدم پشتیبانی نسخه‎های جدید سیستم‎عامل:** سیستم‎عامل‎های تلفن‎همراه به سرعت در حال بروزرسانی هستند، اما بروزرسانی کرونا، به مراتب آهسته‎تر است و همین باعث شده است تا اجرای بازی بر روی نسخه 10 سیستم‎عامل اندروید دارای مشکل باشد.
* **متن‎باز شدن موتور:** موتور بازی‎سازی کرونا به تازگی اعلام کرده است که قصد دارد به صورت متن‎باز به کاربران ارائه شود. این مسأله، سرنوشت این موتور را نامعلوم کرده و باعث می‌شود تا آینده شرکت با خطر روبه رو شود، زیرا مشخص نیست که در آینده، آیا توسعه‎دهندگان، از این مسأله استقبال خوبی خواهند کرد یا خیر.
* **زبان برنامه‌نویسی لوا:** زبان برنامه‎نویسی لوا علاوه بر فایده، معایبی نیز دارد، که می‎توان به عدم پشتیبانی از شی‎گرایی اشاره کرد که در برنامه‎های توسعه‌یافته و عظیم، شی‎گرایی رویکرد مناسبی برای برنامه‎نویسی است.
* **انجمن[[20]](#footnote-20) ضعیف:** این موتور، انجمن ضعیفی دارد و برنامه‌نویسان شرکت برای رفع مشکلات و جست‌وجو با مشکل مواجه می‎شدند.

### 3-1-2- موتور یونیتی

با مقایسه مزایا و معایب موتور بازی‌سازی کرونا، متوجه شدیم که باید پلتفرمی انتخاب شود که با همراه داشتن مزایای موتور کرونا، مشکلات آن موتور را نیز رفع کند. و در بین گزینه‎های موجود، موتور بازی‎سازی یونیتی انتخاب ما بود. دلایل انتخاب این موتور را در ادامه ذکر می‎کنیم:

* **پشتیبانی از چند پلتفرم:** یکی از دلایل موفقیت یونیتی، اجرا و پیاده‌سازی آن بر روی پلتفرم‌های مختلف است. برنامه‌ای که بر روی یک پلتفرم نوشته فقط با چند کلیک برای استفاده در یک پلتفرم دیگر آماده خواهد شد. یونیتی هم‌اکنون پلتفرم‌های تلفن‎همراه، کامپیوتر، وب و کنسول‌های بازی را پشتیبانی می‌کند. برای نمونه:
* بر روی تلفن‎همراه، سیستم‎های اندروید، بلک بری، ویندوز و iOS
* بر روی کامپیوتر‎ها، سیستم‌عامل‌های لینوکس، ویندوز و مک
* برای اهداف وب، Flash، Web Playerو Google Client

پشتیبانی می‌شوند.

* **فریم بر ثانیه:** این موتور نیز مانند کرونا تمامی ابعاد صفحه را پشتیبانی می‎کند.
* **شبیه‎ساز:** یونیتی شبیه‎سازی برای تمامی پلتفرم‌ها دارد و استفاده از آن بسیار آسان است.
* **بروز بودن:** یونیتی اکثرا افزونه‎ها را دارد و به سرعت به‌روزرسانی‎ می‌شود که مشکلات موتور کرونا را برطرف می‎کند.
* **زبان برنامه‌نویسی سی‌شارپ:** زبان برنامه‌نویسی استفاده شده در این موتور، سی‎شارپ است که شباهت نحوی[[21]](#footnote-21) به زبان جاوا دارد. همچنین این زبان از ساختار شی‎گرایی پشتیبانی می‌کند.
* **انجمن قوی:** یونیتی یکی از بهترین موتورهای موجود است که امکانات و ابزار های دو بعدی و سه بعدی زیادی را دراختیار کاربران قرار می‎‌دهد. در حال حاضر 34% از بهترین بازی‌های تلفن‎همراه و 90% از بازی‌های واقعیت مجازی توسط یونیتی ساخته شده‌اند.

## 3-2- تسلط بر کد

قدم بعدی پس از انتخاب پلتفرم یا به عبارتی مهم‎ترین قدم، تسلط بر کد سابق است. تسلط بر کد سابق و همچنین محیط بازی برای پیاده‌سازی نسخه جدید ضروری است، زیرا هم باعث می‌شود تا با راهکارهای توسعه‎دهندگان سابق برای حل مشکل آشنا شویم و هم از قسمت‎های پنهان بازی اطلاع یابیم. برای این منظور در ابتدای امر شروع به یادگیری زبان برنامه‌نویسی لوا کردیم. این زبان، زبانی اسکریپتی است و آشنایی بنده با زبان پایتون، کمک شایانی برای یادگیری زبان کرد.

برای تمرین این زبان، اقدام به پیاده سازی صفحه‎ای از بازی گروهی کردیم. این وظیفه جامع و کامل بود و تقریبا تمام قسمت‎های view را پوشش می‎داد. پس از تسلط نسبی بر زبان لوا، به باقی مراحل پرداختیم.

## 3-2- مستند سازی

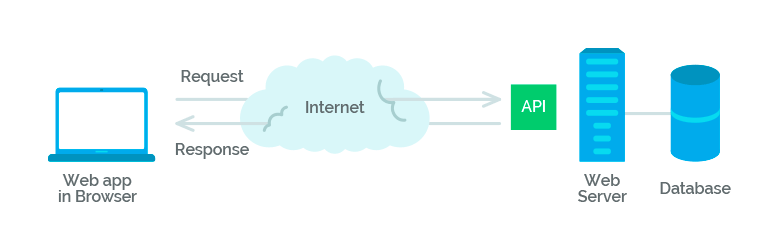
همان‎گونه که در قسمت مقدمه گزارش کارآموزی ذکر کردیم، بازی تحت شبکه و به صورت مشتری/سرویس‎دهنده پیاده‎سازی شده است. مشکلی که در پیاده‎سازی وجود دارد، نبودن مستندی برای کد سمت مشتری و همچنین برای درخواست‌ها[[22]](#footnote-22) و پاسخ‎ها[[23]](#footnote-23) است. چالش بعدی ما پیدا کردن ابزار مناسب جهت مستندسازی بود.

برای مستندسازی کد سمت مشتری، ابتدا نیاز به نوشتن نیازمندی‎ها بود. برای این نیازمندی‎ها در ابتدا برای هر قسمت از بازی، داستانی[[24]](#footnote-24) نوشتیم و در هر قسمت تمامی درخواست‌ها و همچنین پیش‌شرایط و اثرات بعدی آن قسمت را نوشتیم تا در مرحله بعدی برای طراحی بازی به دیدی کلی از بازی رسیده باشیم.

در قسمت درخواست‎‌ها، برای اتصال به سرویس‌دهنده از Restful API استفاده شده بود. که در ادامه به توضیح آن می‌پردازیم[4].

### 3-2-1- API

[[25]](#footnote-25)API که به رابط برنامه‌نویسی کاربردی ترجمه می‌شود، یک مجموعه از قواعد و مکانیزم‌ها است که از طریق آن برنامه‎‌های کاربردی و یا بخش‎های مختلف برنامه با همدیگر ارتباط برقرار می‌کنند. نام خود این مکانیزم، بیانگر همه چیز است. منظور از رابط مفهومی ا‌ست که دو شئ یا دو موجودیت مختلف را به همدیگر مرتبط می‌کند. API می‌تواند داده‌هایی که شما برای برنامه‎کاربردی خود نیاز دارید را از طریق یک فرمت مناسب به خروجی بفرستد و یا آن‌ را برگشت دهد. فرمت JSON و XML از این دست فرمت‌ها هستند. شکل 1-3 نمای یک API را نشان می‎دهد.



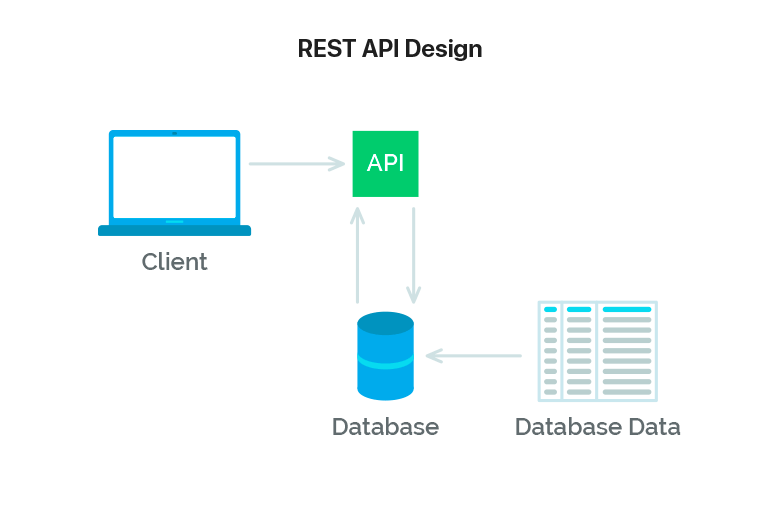
شکل 1-3- نمای یک API

### 3-2-2- REST

این کلمه اختصاری برای Representational State Transfer است. تعریف REST را به صورت ساده می‌توان اینطور بیان کرد: نمایش اطلاعات برای کاربران از راهی که خوانایی بالایی داشته باشد. یکی از مفاهیم اصلی که باید در ارتباط با REST بدانید این است که REST یک پروتکل یا استاندارد نیست، این تنها یک راه‌حل و یا یک سبک معماری برای نوشتن APIها است.

### 3-2-3- RESTful API

حال که فهمیدیم REST چیست، درک کردن RESTful API بسیار ساده‌تر است. همانطور که گفته شد REST یک روش معماری و چیدمان است و RESTful را می‌توان مفسری برای REST دانست. برای مثال اگر شما یک سرویس‌دهنده دارید و قسمت Back-End برنامه، یک REST API دارد. اگر یک کاربر از سمت مشتری[[26]](#footnote-26)، یک درخواست برای استفاده از API بکند، کاربر شما Restful خواهد بود. شکل 2-3 معماری یک REST API را نشان می‎دهد.



شکل 2-3- معماری یک REST API

## 3-3- زبان YAML

پس از فهمیدن مفهوم RESTful API، ما نیاز به مستندسازی درخواست‌ها و پاسخ‌ها داشتیم. چالش بعدی ما پیدا کردن ابزاری سریع، ساده و قابل استفاده بود. ابزاری که ما برای مستندسازی انتخاب کردیم، زبان برنامه‌نویسی YAML بود.

YAML یک زبان برای طبقه‌بندی داده قابل خواندن برای انسان است که معمولاً برای فایل‌های پیکربندی استفاده می‌شود، اما می‌تواند در کاربردهای دیگری هم مورد استفاده قرارگیرد[5]. YAML بسیاری از برنامه‌های ارتباطات مشابهی را به عنوان XML هدف قرار می‌دهد. این زبان به دو روش نوشته می‌شود:

1- سبک زبان پایتون که با پیش‌آمدی در متن، تودرتویی را پشتیبانی می‌کند.

2- روش فشرده‌تری که مانند JSON برای لیست‌ها از [] و برای نگاشت[[27]](#footnote-27)‌ها از {} استفاده می‌کند که YAML 1.2 را زیر مجموعهی JSON قرار می‌دهد.

انواع داده‌های شخصی‌سازی شده در YAML مجاز هستند، اما YAML به صورت طبیعی اعداد (مانند رشته‌ها، اعداد صحیح و شناور)، لیست‌ها و آرایه‌ها (که رمزنگاری‌شده یا نگاشت شناخته می‌شود) را رمز گذاری می‌کند.

پسوند نام پرونده برای YAML، از سال ۲۰۰۶ تا به اکنون، yaml. است.

## 3-4- یادگیری یونیتی

برای یادگیری یونیتی، با دو بخش مهم مواجه بودیم:

- یادگیری زبان سی‌شارپ.

- یادگیری یونیتی.

به علت تسلط بنده بر زبان جاوا، در یادگیری زبان سی‌شارپ مشکلی نداشتم.

یونیتی شامل سه قسمت می‌شود، یونیتی UI، 2D و 3D.

برای این بازی، ما احتیاج به یادگیری Unity UI داشتیم. برای یادگیری این مفاهیم از دوره‎های برخط یودمی[[28]](#footnote-28) استفاده کردیم.

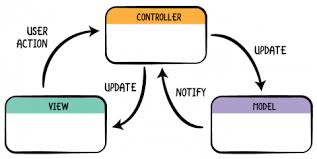
## 3-5- طراحی

به علت تفاوت ساختار زبان‎های برنامه‎نویسی دو موتور و همچنین کمبود وقت در طراحی نسخه اولیه کوییزآو کینگز، مجبور به طراحی دوباره کلاس‎ها شدیم. معماری انتخابی ما، معماری [[29]](#footnote-29)MVC بود، البته یونیتی معماری [[30]](#footnote-30)ECS را نیز برای بازی‎ها پیشنهاد می‌کند، که با توجه به شباهت بازی به برنامه‎کاربردی تلفن‎همراه از آن معماری استفاده نکردیم و به همین جهت به توضیح آن نمی‎پردازیم. در طراحی کلاس‎‌ها سعی شد از مفاهیم شی‎گرایی و الگوهای طراحی[[31]](#footnote-31) استفاده شود.

### 3-5-1- معماری MVC

MVC نوعی روش معماری نرم‌افزار است که در توسعه‌ی وب اپلیکیشن‌ها بسیار پرکاربرد است و ورود آن به صنعت توسعه‌ی نرم‌افزار به دهه‎ی 1970 بازمی‌گردد. امروزه چارچوب[[32]](#footnote-32)‌های مطرحی که در توسعه‎ی نرم‌افزارهای کوچک و بزرگ مورد استفاده قرار می‌گیرند مبتنی بر این معماری‌اند.

Trygve Reenskaug  یک دانشمند علوم کامپیوتری نروژی است که به عنوان خالق اصلی معماری MVC شناخته می‌شود. وی در سال 1979 به منظور توسعه‌ی [[33]](#footnote-33)GUI با استفاده از زبان Smalltalk از این معماری استفاده کرد و در سال‌های بعد این معماری به مرور در میان توسعه‌دهندگان مختلف رواج پیدا کرد تا جایی که الگوهای دیگری از جمله HMVC ،MVA ،MVP و MVVM بر این اساس شکل گرفتند[6]. شکل 3-3 نمایی از یک معماری MVC را نشان می‎دهد.



شکل 3-3- نمایش معماری MVC

### 3-5-2- Model

بخش Model را به نوعی می‌توان به منزله‎ی مغز برنامه‌کاربردی در نظر گرفت به طوری که اصطلاحاً Business Logic یا به عبارتی «آنچه اپلیکیشن به خاطرش توسعه یافته است» در این لایه طرح‌ریزی می‌شود. مسلماً نیاز به توضیح نیست که مثلاً در یک وب اپلیکیشن بخشی از منطق نرم‌افزار مرتبط با ارتباط با پایگاه‎داده[[34]](#footnote-34) به منظور انجام عملیات [[35]](#footnote-35)CRUD است که وظایفی از این دست در Model عملی می‌گردند.

Controller که در ادامه بیشتر با مفهومش آشنا خواهیم شد، همواره ارتباط تنگاتنگی با مدل دارا است به طوری که می‌توان گفت ارتباط Model با View از طریق Controller امکان‌پذیر خواهد بود.

### 3-5-3- View

بخش View، همان‌طور که از نام آن مشخص است، این وظیفه را دارا است تا داده‎هایی که در Model ساخته و پرداخته شده را در معرض دید کاربران وب‎اپلیکیشن قرار دهد و به عبارتی می‌توان گفت که همان User Interface یا به اختصار UI است.

### 3-5-4- Controller

بخش Controller در این معماری سه‌لایه نقش واسط را دارا است به طوری که درخواست‎ها را از بخش View گرفته و در اختیار Model قرار می‌دهد و پس از آنکه Model پردازش‌هایش را روی درخواست ورودی انجام داد، پاسخ را مجدداً در اختیار Controller قرار داده و Controller هم پاسخ نهایی را در اختیار View می‌گذارد تا در معرض دید کاربران قرار دهد.

## 3-6- آکادمی

در طی کارآموزی، جلساتی با عنوان آکادمی نیز برقرار می‎شد، هدف این جلسات افزایش سطح دانش فنی و تیمی بود. از مواردی که در جلسات آکادمی بررسی شد، می‎توان به موارد زیر اشاره کرد:

* آشنایی با الگوهای طراحی singleton و command.
* چگونه یک merge request خوب ایجاد کنیم؟
* آشنایی با اصول Clean Code.
* آشنایی با مفاهیم UML[[36]](#footnote-36).

جلسات به صورت ارائه‎ی شخصی در مورد موضوعی بود. پس از آن افراد تیم به بحث و گفت‌وگو درباره‎ی آن موضوع می‌پرداختند. بنده نیز در این ارائه‌‎ها شرکت داشته و مفاهیم UML را برای افراد حاضر در تیم، ارائه دادم.

## 3-7- متدولوژی اسکرام

برای چابک[[37]](#footnote-37) شدن و مدیریت چابک پروژه‌ها روش‌های مختلفی وجود دارد، برخی از روش‌های مدیریت پروژه‌های چابک به نسبت سایر آنها معروف تر هستند و بیشتر مورد توجه و استفاده قرار گرفته‌اند. آشنایی کامل، بی نقص و همزمان با همه روش‌های مدیریت چابک پروژه مانند اسکرام[[38]](#footnote-38)، کانبان[[39]](#footnote-39) و XP امری ضروری نیست اما داشتن آشنایی با آنها در اجرای چابکی به شما این امکان را می‌دهد تا در انتخاب روش‌های متناسب با هر نیاز و شرایط انتخاب‎های متعددی داشته باشید.

اسکرام پر استفاده ترین روش در مدیریت چابک پروژه‌ها است که سه رکن اصلی دارد[7]:

1. شفافیت[[40]](#footnote-40): ایجاد دید روشن برای افرادی که در به وجود آمدن نتیجه مسئول هستند.
2. بازبینی[[41]](#footnote-41): بررسی مستمر اینکه در حال انجام چه کاری با توجه به اهداف هستید.
3. تطبیبق[[42]](#footnote-42): تغییر فرآیند‌های تیم به منظور کاهش خطاها.

تیم‌های اسکرام در بسته‌های کوتاه‌مدت تکراری به نام اسپرینت[[43]](#footnote-43) کار می‌کنند[8]. در عمل، اسکرام از تعدادی از فعالیت‌ها و رسم‌های تکرارشونده در هر اسپرینت پیروی می‌کند. با این کار در واقع اسکرام گاردریل‌[[44]](#footnote-44)های محافظی ایجاد می‌کنند تا افراد به طور موثرتری کار کنند.

# فصل چهارم نتیجه گیری

## 4-1- جمع بندی

هدف این کارآموزی، اقدام برای تغییر پلتفرم یک برنامه توسعه یافته و عظیم بود. این کار سختی‎های بسیاری به همراه داشت و همچنین خطر شکست همیشه در کمین بود. اما تیم توسعه دهنده کوییزآو کینگز این خطرات را به جان خریدند تا به محصولی بهتر دست پیدا کنند. مواجه شدن با موتور بازی سازی جدید و بدون دانش قبلی، کاری پر خطر بود که تیم ما توانست با تلاش بی‎وقفه گام بزرگی را در این زمینه بردارد.

در هنگامی که این گزارش نوشته می‌شود، نسخه اولیه بازی به زبان انگلیسی در کشور فیلیپین در حال انتشار و اجرا است.

## 4-2- کارهای آینده

کار تغییر پلتفرم به علت حجم بزرگ بازی هنوز به صورت کامل انجام نشده است، قصد ما ادامه این کار تا رسیدن به نسخه کاملی از بازی است و در این راه، روش‎های متفاوتی را برای بهتر شدن بازدهی بازی به کار خواهیم برد.

# منابع و مراجع

[1] “همه چیز دربارة بازی های رایانه ای.” https://hawzah.net/fa/Magazine/View/4892/7627/95546/همه-چیز-دربارة-بازی-های-رایانه-ای (accessed Sep. 24, 2020).

[2] “تفاوت سرور (Server) و کلاینت (Client).” https://saprasanat.net/تفاوت-سرور-server-و-کلاینت-client/ (accessed Sep. 25, 2020).

[3] “آیا Corona SDK بهترین پلت فرم برای توسعه بازی های موبایل است؟ -.” http://www.wiadevelopers.com/2018/10/18/آیا-corona-sdk-بهترین-پلت-فرم-برای-توسعه-بازی-ها/ (accessed Sep. 25, 2020).

[4] “RESTful API به زبان ساده چیست؟.” https://roocket.ir/articles/a-beginners-tutorial-for-understanding-restful-api (accessed Sep. 25, 2020).

[5] “YAML چیست؟ :: تک بست - فناوری با طعمی متفاوت.” http://techbest.blog.ir/1397/05/28/YAML-چیست (accessed Sep. 25, 2020).

[6] “معماری MVC چیست؟.” https://sokanacademy.com/courses/mvc/10230/معماری-ام-وی-سی-چیست (accessed Sep. 25, 2020).

[7] “اصول اولیه متدولوژی اسکرام چیست؟ - ویرگول.” https://virgool.io/@reza.sabourinejad/اصول-اولیه-متدولوژی-اسکرام-چیست-ebo9xfcv0t84 (accessed Sep. 25, 2020).

[8] “اسکرام چیست و چگونه آن را استفاده کنیم؟ | شبکه.” https://www.shabakeh-mag.com/workshop/programming/13799/اسکرام-چیست-و-چگونه-آن-را-استفاده-کنیم؟ (accessed Sep. 25, 2020).

1. Open source [↑](#footnote-ref-1)
2. Android [↑](#footnote-ref-2)
3. Corona Engine [↑](#footnote-ref-3)
4. Unity [↑](#footnote-ref-4)
5. Chat [↑](#footnote-ref-5)
6. Quiz of Kings [↑](#footnote-ref-6)
7. Client [↑](#footnote-ref-7)
8. Server [↑](#footnote-ref-8)
9. Server [↑](#footnote-ref-9)
10. Game Server [↑](#footnote-ref-10)
11. Task [↑](#footnote-ref-11)
12. Chat [↑](#footnote-ref-12)
13. Google Store [↑](#footnote-ref-13)
14. Online [↑](#footnote-ref-14)
15. Applications [↑](#footnote-ref-15)
16. Multi platform [↑](#footnote-ref-16)
17. Frame per Second (FPS) [↑](#footnote-ref-17)
18. Software development kit [↑](#footnote-ref-18)
19. Plugin [↑](#footnote-ref-19)
20. Community [↑](#footnote-ref-20)
21. Syntactically [↑](#footnote-ref-21)
22. Requests [↑](#footnote-ref-22)
23. Responses [↑](#footnote-ref-23)
24. User story [↑](#footnote-ref-24)
25. Application Programming Interface [↑](#footnote-ref-25)
26. Client Side [↑](#footnote-ref-26)
27. Map [↑](#footnote-ref-27)
28. <https://www.udemy.com/> [↑](#footnote-ref-28)
29. Model-View-Controller [↑](#footnote-ref-29)
30. Entity-Component-System [↑](#footnote-ref-30)
31. Design patterns [↑](#footnote-ref-31)
32. Framework [↑](#footnote-ref-32)
33. Graphical User Interface [↑](#footnote-ref-33)
34. Database [↑](#footnote-ref-34)
35. سرواژه‎ی **CRUD** برگرفته از کلمات **U**pdate ،**R**ead ،**C**reate و **D**elete است که به ترتیب به منظور «ثبت»، «فراخوانی»، «به‌روزرسانی» و «حذف» داده‌ها از دیتابیس مورد استفاده قرار می‌گیرند. [↑](#footnote-ref-35)
36. Unified Modeling Language [↑](#footnote-ref-36)
37. Agile [↑](#footnote-ref-37)
38. Scrum [↑](#footnote-ref-38)
39. Kanban [↑](#footnote-ref-39)
40. Transparency [↑](#footnote-ref-40)
41. Inspection [↑](#footnote-ref-41)
42. Adaptation [↑](#footnote-ref-42)
43. Sprint [↑](#footnote-ref-43)
44. Guardrails [↑](#footnote-ref-44)