پروژه درس برنامه نویسی پیشرفته

زمستان ۱۳۹۸ - دانشکده علوم ریاضی
دانشگاه صنعتی شریف

تیم طراحی

امیرحسین ابراهیمی

فیروزه ابریشمی

مهدی استاد شریف معمار

امیررضا اکبری

محمدجواد اکبری

زهرا جانعلی زاده

علیرضا خادم

امیر محمد شعبانی

ساحل مس فروش

فرزین نصیری

فهرست عنوان‌ها

معرفی پروژه

پروژه شما پیاده سازی بازی hearthstone است و شما می‌توانید توضیحات بازی را در صفحات بعدی مطالعه کنید.

فایلی که در حال مطالعه آن هستید، شامل توضیحات کلی هر بخش از پروژه شما است و ماهیت آنها را مشخص میکند. لطفا آن را تا انتها مطالعه کنید تا نسبت به پروژه خود شناخت کاملی داشته باشید. در زمان منتشر شدن هر بخش (فاز) پروژه یک فایل توضیحی کامل همراه با جزئیات نمره‌دهی و ارزیابی، راهنمایی و نمرات امتیازی و ... به شما داده می شود. هر قسمت از پروژه ممکن است توسط افراد متفاوتی طراحی شده باشد که در فایل توضیحی هر بخش مشخص می شود.

اهداف پروژه

- شناخت پیدا کردن نسبت به توسعه نرم افزار (Software development)
- آشنایی با گرافیک کامپیوتری و طراحی رابطهای کاربری
- آشنایی با معماری‌های طراحی نرم افزارهای کاربردی، برنامه نویسی شی گرا و تمیز کد زدن
- شناخت ویژگی‌های اختصاصی زبان جاوا
- کار با فایل‌ها
- آشنایی با پروتکل‌ها شبکه و استفاده از آن‌ها
- استفاده از پایگاه‌های داده و ذخیرسازی درست اطلاعات
- کار با Reflection جاوا

رفع اشکال

شما می‌توانید برای رفع پرسش‌ها، اشکال‌ها و ابهام‌های خود را از تیم طراحان پروژه در میان بگذارید و برطرف کنید.

ارزیابی

- پروژه شما به پنج بخش تقسیم شده که هر بخش(یا هر چند بخش) به طور جداگانه ارزیابی و به صورت حضوری تحویل گرفته می شود.
- هر بخش پروژه باید حداکثر تا تاریخی که به شما اعلام شده در کوئرا آپلود شود. کدی که در زمان تحویل مورد ارزیابی قرار می‌گیرد، کد آپلود شده در کوئرا می باشد. دقت کنید که ددلاین های اعلام شده قابل تغییر نمی‌باشند پس برنامه ریزی لازم را برای رساندن پروژه خود به ددلاین ها داشته باشید.
- دقت کنید که فقط کارکردن کد مدنظر نیست و از شما انتظار می رود که به صورت اصولی، تمیز و پیشرفته کد بزنید!
- بارم بندی هر بخش به همراه جزئیات پیاده سازی در زمان های مشخص شده به شما اطلاع داده می شود.
- در هر بخش پروژه شما ملزم به نوشتن یک فایل توضیحی کامل در مورد کد خود هستید. روش نوشتن این فایل توضیحی در صفحه بعدی نوشته شده است.

نوشتن بخش توضیحات

هر بخش (فاز) پروژه شما نیاز به یک فایل توضیحی است که موارد زیر را شامل می شود:

1. منابع استفاده شده برای پیاده سازی کد شامل
   a. منبع تصاویر و ...
   b. منبع کدها و ...
   c. مشورت های انجام شده
   d. کتابخانه های استفاده شده
2. روش کارکرد کد شما به همراه نقاط قوت و ضعف آن
3. ارائه دلیل برای انتخاب هایی که انجام داده اید (مثلاً چرا از یک کتابخانه خاص یا طراحی خاص استفاده کرده اید)

این نوشته به ارزیابی سریع تر و راحت تر پروژه شما کمک فراوانی می کند. ترجیحا توضیحات پروژه کوتاه، مختصر و مفید باشد.

اتفاق‌های غیر مجاز

موارد غیر مجاز

- عدم تسلط کافی بر کد پروژه
- شباهت بیش از حد دو یا چند پروژه
- واگذاری کامل یا بخشی از پروژه به شخصی دیگر

رخ دادن این اتفاق‌ها برای هیچ فردی قابل پذیرش نیست و در صورت بروز هر کدام از این اتفاق‌ها ممکن است هر تصمیمی در رابطه با ارزیابی فرد گرفته شود.

در صورتی که یکی از این اتفاق‌ها رخ داده باشد لازم است که افراد حتما دلیل این مساله را پیش از تشخیص توسط تیم درس اعلام کنند و دلیل این اتفاق را توضیح دهند. در این صورت فقط ارزیابی مربوط به بخش اعلام شده تحت تاثیر قرار خواهد گرفت.

در صورتی مشاهده یکی از این اتفاق‌ها توسط تیم درس و پیش از اعلام فرد رخ دهد این تیم از فرد درخواست خواهد کرد که در این رابطه توضیح دهند و در صورت قابل قبول نبودن توضیح فرد موفق به گذراندن درس نخواهد شد.

توضیح پروژه

آشنایی کلی با بازی

بازی hearthstone یک بازی کارتی استراتژیک بوده که توسط شرکت Blizzard Entertainment در سال 2014 میلادی ساخته شده و تاکنون چندین نسخه از این بازی عرضه شده است. این بازی شباهت زیادی به بازی های کارتی دیگر دارد. در این بازی شانس و استراتژی دست به دست هم می دهند تا تجربه ای لذت بخش برای بازیکنان آن به وجود بیاورند.

هم‌اکنون این بازی روی پلتفرم های iOS, Android, Microsoft Windows, Mac OS قابلیت اجرا دارد.

تریلر بازی

https://www.aparat.com/v/149Nm

ویدیو از گیم پلی نسخه‌ای از بازی

https://www.aparat.com/v/te5qn

سایت سازنده بازی اصلی

https://playhearthstone.com/en-us/blog

یکی از صفحات طرفداران بازی

https://www.hearthpwn.co

ماهیت بازی

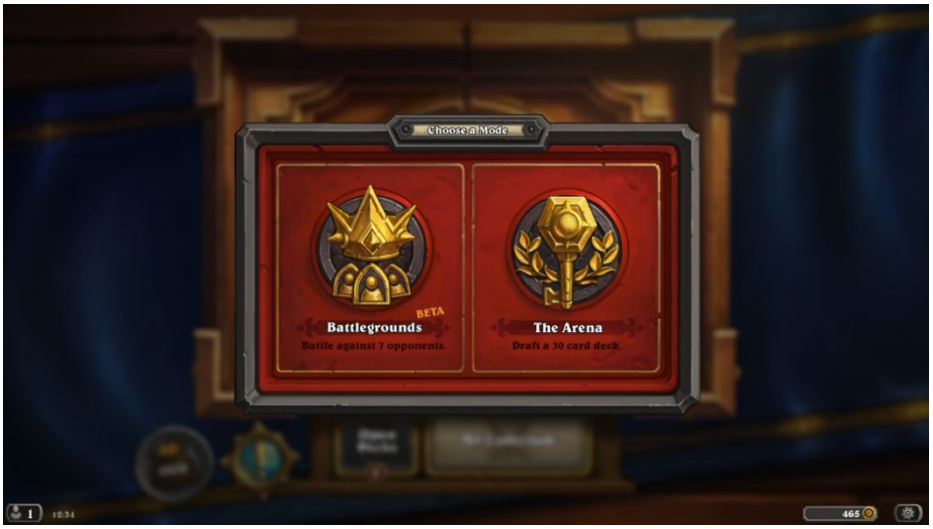
منو بازی

پس از ورود به اکانت با صفحه بالا روبرو می‌شوید.

قسمت play همان بازی دو نفره ساده بوده که به صورت آنلاین با شخص دیگری بازی می‌کنید.

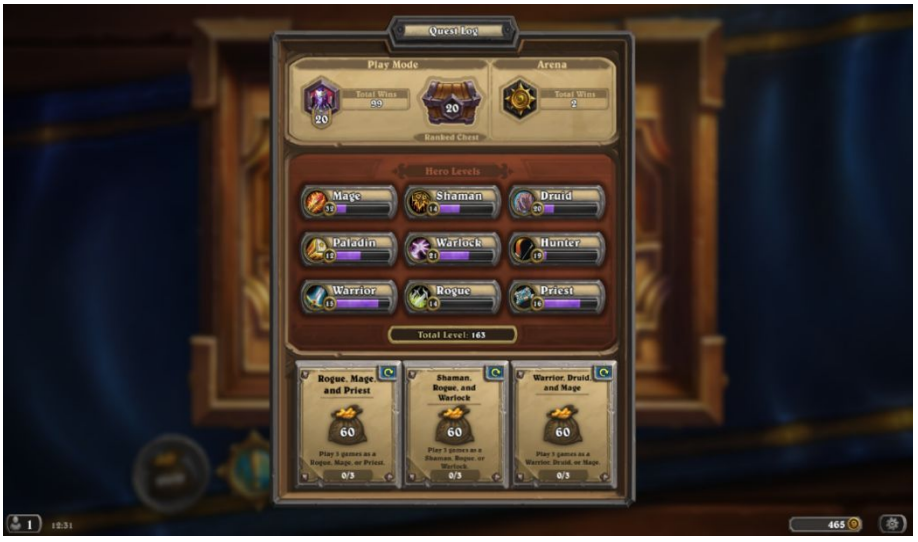
قسمت Solo Adventure بخش آفلاین آن بوده که با Boss هایی که توسط سازنده بازی طراحی شده بازی می‌کنید و با بردن در آنها جایزه می‌گیرد.

قسمت Tavern Brawl بخش خاصی است که همواره در دسترس نیست اما وقتی باز می شود، شما می توانید شکلی جدید از بازی با قوانین متفاوت و ... را تجربه کنید.



قسمت Modes دارای دو بخش Arena و Battleground بوده که شامل بازی های رقابتی و چندنفره می باشد.

در قسمت Shop نیز شما میتوانید یک Pack خریداری کنید و در قسمت Open Packs آن را باز کرده آن گاه به صورت تصادفی به آن 5 کارت اختصاص داده میشود که حتما باید یکی از آن کارت ها از نوع کمیاب یا بالاتر باشد. (انواع کارت ها در فایل مربوط به بخش اول پروژه برای شما توضیح داده می شود.)



قسمت Quest که با شکل علامت تعجب مشخص شده با انجام دادن ماموریت‌های گوناگون به شما سکه میدهد که میتوانید با آن Pack خریداری کرده. همچنین در این قسمت مشخصات کلی از تعداد بازی‌های برده و Level قهرمان ها به شما داده میشود.

قسمت My Collection تمام کارت هایی که شما دارید را نشان میدهد که در دسته بندی مخصوص خود قرار دارند و در سمت راست میتوانید Deck مخصوص به خود را بچینید تا با آن بازی کنید.

زمین بازی

- در پایین صفحه قهرمان شما قرار دارد. زیر آن کارت هایی که در دست شما قرار دارد نشان داده شده است.
- در قسمت بالا قهرمان دشمن قرار دارد که جان آن و تعداد کارت هایی که در دست آن قرار دارد مشخص است.
- با زدن دکمه end turn به بازی اعلام میکنید که دیگر قرار نیست تا turn بعد کاری انجام دهید و turn را به دشمن می‌دهید.
- دسته کارت‌هایی است که برای آن بازی چیده‌اید. (deck خود را در بخش my collection ساخته اید.)
- در این بخش تعداد ماناهایی که در دسترس دارید را نمایش میدهد. مانا به نوعی پول شما برای استفاده کردن از کارت ها در هنگام بازی کردن است. برای هر کارت باید مقدار مختص به آن را پرداخت کنید.
- این نیرو یک نیرو خاص است که برای هر قهرمان متفاوت است که به آن هیرو پاور(hero power) می‌گویند.
- هر قهرمان مقداری جان به خصوص خود را دارد.

- قسمت زمین بازی است که اگر میخواهید کارتی را بازی کنید باید کارت را به این طرف بکشید. کارت‌های دشمن در نصفه بالا قرار میگیرد و کارت‌های شما در نصفه پایین.
- در این قسمت تمام اتفاقات بازی ضبط شده و به ترتیب نمایش داده می‌شود.

کارت‌ها

1 – تعداد مانای مورد نیاز برای بازی کردن این کارت را نشان میدهد. در حقیقت مانا‌ها مانند سکه هایی هستند که در هر turn فقط به اندازه آن میتوانید خرج کنید.

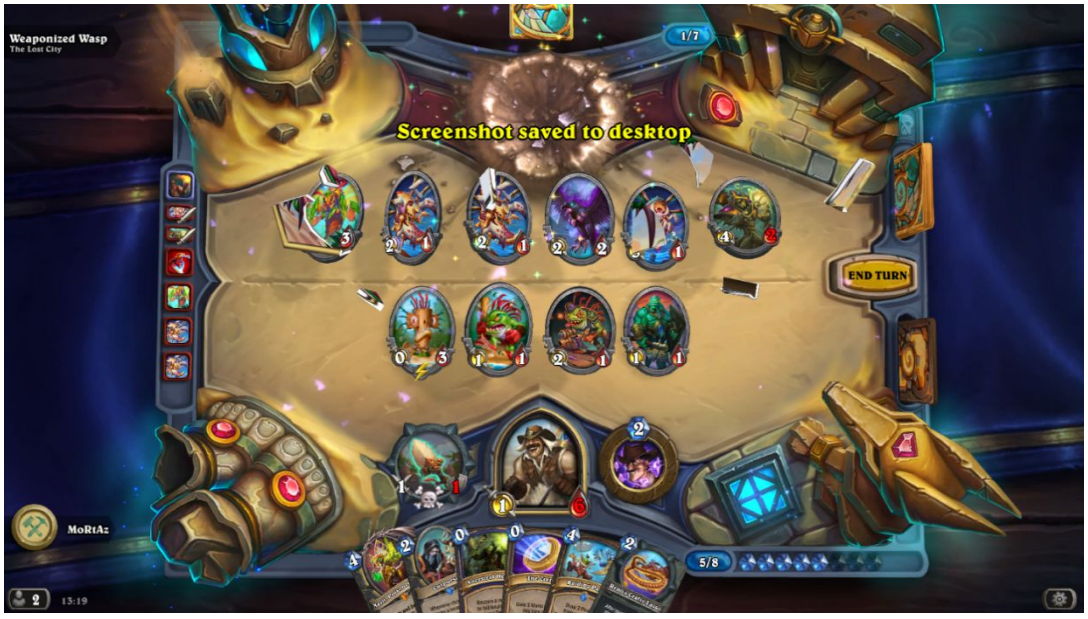
2 و 3 – اگر کارتی که شما قرار است بازی کنید از نوع اسپل (spell) باشد این دو مورد را ندارد و در زمین کاشته نمیشود و فقط کاری که قرار است انجام دهد را انجام داده و ناپدید می‌شود . در غیر اینصورت مورد 2 میزان damage ای که به دشمن میدهد را نشان میدهد.(در صورت حمله آن) و مورد 3 میزان جانی که دارد.(توجه کنید که جان آن هم در زمان حمله و هم در زمان دفاع ممکن است کاسته شود)

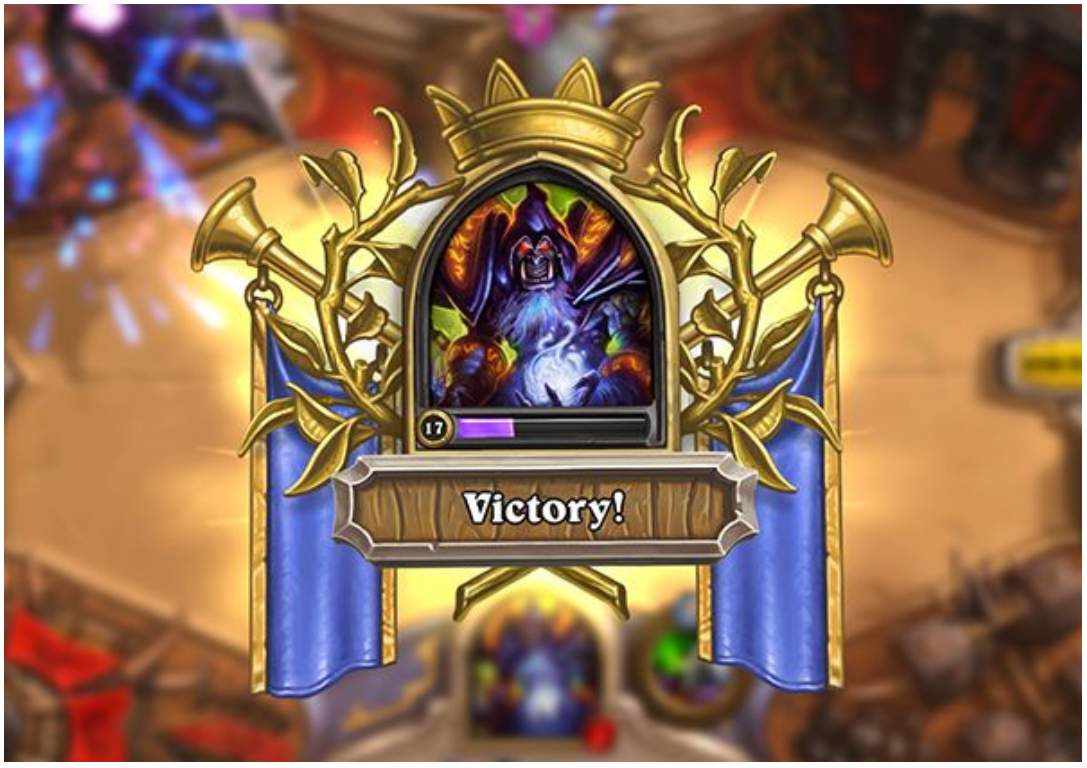
4 – درآن قسمت اسم کارت و عکسی از کارت و قسمت کریستالی آن نوع کمیابی آن را نشان میدهد به طور مثال این کارت از نوع رایج است.

5 – در قسمت پایین نوع کارت که مینیون است را نمایش میدهد و در قسمت وسط توضیحاتی در مورد کاری که این مینیون در شرایط خاص انجام میدهد به ما میگوید.

پایان بازی

بازی تا جایی ادامه پیدا خواهد کرد تا جان یکی از قهرمان ها صفر شود و یا بازیکنی انصراف دهد. سپس حریف وی برنده اعلام می شود.





بخش‌های پروژه

بخش اول - Game Models

در برنامه نویسی، اشیا دنیا واقعی (چه فیزیک و چه غیر فیزیکی) را با کلاس ها، ساختار ها و ... نشان می دهیم که کاملا وابسته به زبان است. با توجه به اینکه هر پروژه نیاز به اشیا متفاوت و زیادی دارد بهتر است همه آن ها به درستی طراحی شده و سپس توسط برنامه نویسان پروژه استفاده شوند.

دیتا مدل ها، اشیا بازی شما هستند که توسط قسمت های مختلف برنامه استفاده می شوند. وظیفه اصلی شما در این بخش طراحی و مدیریت این مدل ها است. ویژگی های اصلی این بخش شامل موارد زیر است:

- ساخت و نگهداری اشیا بازی(کارت‌ها، قهرمانان و ...)
- پیاده سازی یک محیط command line interface برای تعامل با بازی
- ذخیره کردن اطلاعات بازی
- سیستم مدیریت بازیکن ها
- ویژگی‌های درون بازی (فروشگاه، مدیریت کارت‌ها و ...)

سیستم مدیریت بازیکن‌ها

در هر بازی وجود گزینه های ثبت نام، ورود، خروج و حذف حساب الزامی است. تمام اطلاعات مرتبط با بازیکن‌ها از جمله (قهرمان‌های باز شده، کارت‌ها، امتیازات و بردها و باخت‌ها و ...) بایستی ذخیره شود و بعد از خروج کامل از بازی قابل بازیابی باشد. وظیفه شما پیاده سازی این سیستم ها به همراه طراحی روش ذخیره سازی آن ها است.

ساخت و نگهداری اشیا بازی

در این بخش شما باید دیتا مدل های بازی را پیاده کنید و سپس با استفاده از ویژگی‌های شی‌گرایی جاوا (شامل وراثت، چند ریختی و ...) و استفاده از الگوها و معماری‌های مناسب این اشیا را مدیریت کنید.

ذخیره سازی اطلاعات بازی و وقایع بازی :

تمام داده های بازی باید به طور جداگانه در فایل هایی ذخیره شوند به طوری که بتوان بدون کامپایل مجدد کد در دیتا مدل های پروژه تغییراتی ایجاد کرد.

ویژگی های درون بازی

بازی شما باید یک فروشگاه برای خرید کارت و قهرمانان جدید داشته باشد. همچین توانایی اضافه کردن کاربرهای دیگر به عنوان دوست را هم باید داشته باشید. هر بازیکن باید بتواند کارت های هر قهرمان خود را مدیریت کند.

رابط کاربری نوشتاری (Command line interface) یعنی یک محیط درون ترمینال یا cmd که با استفاده دستوراتی از پیش تعین شده به ما اجازه تعامل با یک برنامه را می دهد. بسیاری از پروژه های برنامه نویسی از همچین سیستمی به جای یک رابط گرافیکی استفاده می کنند. در این بخش شما باید همچین محیطی را برای تعامل با بازیکنان پیاده سازی کنید. دستوراتی که باید پیاده سازی کنید و جزئیات آن در آینده به شما داده می شود. به عنوان نمونه:

- **Command: ls –players**
  - Usage: Prints all players ever signed up in the game
- **Command: add [cart name/id]**
  - Usage: Adds a card to current hero of the player
- **Command: ch [hero name/id]**
  - Usage: Chooses hero for battle

بخش دوم - Game Client

در دنیا برنامه نویسی معمولا سعی می شود که دنیای گرافیک از دنیا منطق و مدل های بازی جدا باشد. چه در طراحی وب و ... البته این عمل همواره به صورت کامل انجام نمی شود. با توجه به این نکته در این بخش شما باید بین این دو بخش تمایز قائل بشوید و سعی کنید ارتباط آن ها را درست پایه ریزی کنید. زیرا در صورت پیاده سازی اشتباه ممکن است در قسمت های دیگر پروژه با مشکلات زیادی مواجه شوید. این که چگونه این عمل را انجام دهید به عهده ی خلاقیت و فکر خودتان است. و هنگام ارزیابی پروژه باید در مورد آن توضیح دهید.

گرافیک و کلاینت

در بخش دوم پروژه شما باید گرافیک پروژه خودتان را نیز پیاده کنید. البته هدف این بخش فقط پیاده سازی کامل گرافیک برنامه نیست بلکه شما باید یک نسخه client بازی را پیاده کنید که فقط شامل menu ها و ... باشد و توانایی انجام یک بازی دو نفره را ندارد. وظایف این قسمت بازی به صورت زیر است:

1. با کاربر تعامل کند.
2. از کاربر (بازیکن) ورودی بگیرد.
3. انیمیشن ها و ... را مدیریت کند
4. با توجه به اطلاعات ورودی شرایط بازیکن (مثلا کارت های وی و ...) را تغییر دهد.

لطفا دقت کنید که:

- شما مجاز به استفاده از کتابخانه های بازی سازی جاوا (از جمله libgdx و lwjgl و ...) نیستید.
- شما می توانید از هر کتابخانه دیگری برای پیاده سازی گرافیک استفاده کنید.
- می توانید assets های بازی را در این لینک پیدا کنید. همچنین می توانید از منابع مختلف اینترنت استفاده کنید. در آینده یک لینک حاوی بعضی از assets های مهم و احتمالا مورد نیاز به شما داده می شود اما قطعا استفاده از خلاقیت خودتان در ارزیابی اهمیت دارد.

رابط گرافیکی شما باید بخش های زیر را شامل شود:

- صفحه ورود و ثبت نام
- صفحه اصلی و منو بازی
- ساختن leaderboard برای بازیکنانی که تا به حال بازی را تجربه کرده اند
- Shop
- صفحه مدیریت کارت ها و قهرمان ها برای هر بازیکن

دقت کنید در بخش های بعدیی پروژه ویژگی‌های دیگری به کل پروژه و طبیعت رابط کاربری آن اضافه می شود پس بهتر است تلاش کنید تا طراحی شما قابل گسترش باشد. همچنین اضافه کردن قسمت هایی مثل **تنظیمات** ، **credits** و ... نمره امتیازی دارد.

جزئیات پیاده سازی بیشتر گرافیک در داک بخش 2 به طور کامل بیان شده اما دست شما برای بروز خلاقیت باز است و می توانید ویژگی های زیادی به گرافیک بازی استفاده کنید. شایان ذکر است که حد تعادلی برای این زیبایی های بصری وجود دارد و صرف داشتن یک پروژه زیبا که معماری درستی ندارد و یا کار نمی کند مورد قبول نیست و هر چند برای افرادی که بر روی جزئیات گرافیکی پروژه خود زمان زیادی صرف کرده اند نمره امتیازی در نظر گرفته میشود اما زیاده روی در گرافیک برنامه بر روی نمره نهایی شما تاثیری نخواهد داشت.

در ارزیابی این بخش از شما انتظار داریم که به نکات طراحی زیبا و جذاب توجه کافی را داشته باشید ولی لزومی به طراحی انیمیشن های پیچیده و رابط های حرفه ای نیست. این نکات را هنگام اضافه کردن بخش های گرافیکی دیگر پروژه خود مدنظر داشته باشید. علاوه بر این در این مرحله از پروژه شما انتظار می رود که :

- یک بازیکن به راحتی بتواند در بازی ثبت نام کند و در صورت ثبت نام وارد آن شود و یا از آن خارج شود
- بازیکن باید بتواند دسته کارت های خود را ویرایش کند و قهرمان خود را تغییر دهد.
- بازیکن باید بتواند از فروشگاه کارت های جدید خریداری کند و از آن ها در دسته کارت خود استفاده کند.
- بازیکن باید بتواند حساب کاربری خود را ویرایش کند

بخش سوم - Offline Game Client

پس از ورود به حالت play در بازی، بازیکن با دو بخش آفلاین و آنلاین برخورد می کنید. در این بخش شما باید حالت آفلاین را پیاده سازی کنید.

معمول است که در بازی های مختلف قسمت تصمیم گیرنده بازی از گرافیک آن جدا شود. به آن قسمتی از بازی که این که تصمیم گیری را می کند منطق بازی می گویند و می توانید آن را به عنوان مغز متفکر بازی در نظر بگیرید ! تا الان شما این سیستم منطقی را با بخش گرافیکی در هم آمیخته بودید اما حالا که باید خود arena بازی را پیاده کنید نیاز است طراحی دیگری را دنبال کنید.

در این بخش، بعد از ورود به حالت آفلاین، دو بازیکن در مقابل هم قرار میگیرند و نبرد شروع می شود !

بازی باید تاحدی که برای شما مشخص می شود مشابه بازی اصلی عمل کند. شما باید بتوانید از کارتها متفاوت خود استفاده کنید و بازیکنان تجربه خوبی از بازی داشته باشند. جزئیات پیاده سازی در آینده به شما اعلام خواهد شد. بدیهی است که انتظار می رود گرافیک مناسب این بخش اعم از خود board بازی به همراه قسمت ها مختلف آن و ... را نیز پیاده کنید.

برای شهود پیدا کردن نسبت به منطق بازی می توانید بخش زیر را بخوانید:

منطق بازی Game Logic

منظور از منطق بازی آن قسمتی از بازی است که ورودی های بازی را کنترل می کند و با توجه به آن ها حالت بازی را تغییر میدهد. می توانید آن را مغز متفکر بازی بدانید! البته خود گرافیک بازی نیز منطق به خصوص خود را دارد و در واقع آن بخش گرافیک که ورودی ها، انیمیشن ها، دکمه ها و ... را کنترل می کند منطق گرافیک/کلاینت بازی است.

Game logic بازی شما باید کارهای زیر را انجام دهد:

1. از گرافیک بازی اطلاعات را بگیرد
2. اطلاعات را پردازش کند
3. بر اساس اطلاعات تصمیم بگیرد
4. پاسخ را به گرافیک برگرداند تا شکل بازی تغییر کند(حالت بازی را عوض کند)

به عنوان مثل شما یک کارت از دست خود بر میدارد و روی زمین قرار میدهید. کلاینت بازی این را نمایش می دهد در حالی که این منطق بازی است که با توجه به دستور شما به بازی، آن کارت را از دست شما خارج می کند و روی زمین قرار میدهد و ...

ثبت لاگ‌های بازی

علاوه بر قسمت های گفته شده در این بخش شما باید سیستم log های بازی را پیاده کنید. معمولا برنامه ها یک یا چند فایل لاگ از فعالیت خود تهیه می کنند:

```
I 09:17:48.4552651 Load AssetBundle - C:/Program Files (x86)/Hearthstone/Data/win/actors0.unity3d, 17039238

I 09:17:48.8387721 Load AssetBundle - C:/Program Files (x86)/Hearthstone/Data/win/sounds0.unity3d, 5295685

I 09:17:48.8547274 Load AssetBundle - C:/Program Files (x86)/Hearthstone/Data/win/sounds1.unity3d, 5051621

I 09:17:48.8577209 Load AssetBundle - C:/Program Files (x86)/Hearthstone/Data/win/sounds2.unity3d, 4533746
```

بازی شما باید بتواند برای هر بازی دو نفره انجام شده یک فایل game_id.game درست کند که در آن تمام رویداد های بازی ذخیره می شوند به نوعی که بتوان از روی آن ها بازی را دوباره به نمایش درآورد. این فایل ها باید شامل اطلاعات جانبی مثل تاریخ و زمان و مدت بازی و ... هم باشند. البته فرمت این لاگ ها به طور دقیق به شما داده خواهد شد. دقت کنید برای رعایت عدالت و پرهیز از خطا از لاگ های ثبت شده در بازی های شما برای بررسی درستی کد استفاده خواهد شد پس لطفا در پیاده سازی این بخش دقت کافی را داشته باشید.

بخش چهارم - Network and Online Game

تا اینجا شما باید یک بازی کامل با توانایی انجام بازی آفلاین و ... را داشته باشید. پروژه شما در این بخش به دو قسمت مجزا تقسیم می‌شود.

- Client
- Server

همانطور که از نام آنها پیداست سرور مانند یک خدمتکار به مشتری خود خدمات ارائه می دهد. اگر بخش‌های قبلی را به درستی پیاده کرده باشید وظیفه شما در این بخش صرفا ایجاد ارتباط با استفاده از socket programming است. در واقع کلاینت شما که همان رابط کاربری و... است باید بتواند تحت پروتکل(قرارداد) تعریف شده ای با سرور شما ارتباط برقرار کند. شما می توانید از قرارداد تعریف شده به وسیله خودتان استفاده کنید و یا از قرار دادهای دیگری مانند json,protobuf... استفاده کنید. در هر حال باید توجیه منطقی و درستی برای استفاده خودتان ارائه دهید و در صورت نیاز می توانید از کتابخانه های ارائه دهنده این پروتکل ها استفاده کنید.

ماهیت سرور برنامه

در بخش یک و دو شما قسمتی از منطق بازی که مرتبط با خود کلاینت بازی بود را پیاده کرده اید. قسمتی از این منطق به همراه دیتا مدل های شما برای سرور و کلاینت مشترک هستند و هر دو می توانند از آن استفاده کنند. البته سرور شما در حالت کلی قرار است از منطق پیاده شده در بخش 3 برای تصمیم گیری استفاده کند که البته برای آن نیاز دارد از model های تعریف شده و ... نیز استفاده کند از سرور شما انتظار می رود که وظایف زیر را به درستی انجام دهد:

- با توجه به ورودی های کاربر(بازیکن) از سمت کلاینت ، واکنش درست را در کمترین زمان نشان دهد. سعی کنید معماری صحیحی برای این کار پیدا کنید.
- سرور شما توانایی مدیریت کردن تمام بازیکنان درون بازی را دارد و میتواند هر تعداد بازی بین دو بازیکن مجزا را به صورت بهینه و سریع (بدون لگ) اجرا کند.
- سرور شما باید fault tolerant باشد. به این معنا که در صورت بروز هرگونه مشکل(خارج شدن بازیکن، از بین رفتن ارتباط بین سرور و یکی یا همه کلاینت ها و ...) سرور به بهترین شکل ممکن خطا را مدیریت کرده و در صورت لزوم به کاربر نشان دهد و او را به صفحه اصلی هدایت کند.
- هر وظیفه دیگری که به کلاینت بازی مربوط نیست را نیز سرور بر عهده میگیرد.
- سیستم cli پیاده شده در بخش یک باید به کلی به سرور برنامه منطق شود. اینگونه شما با کامند لاین یا ترمینال خود میتوانید با سرور خود ارتباط برقرار کنید و به عنوان کاربر admin در به بازی ها دسترسی داشته باشید.

ماهیت کلاینت برنامه

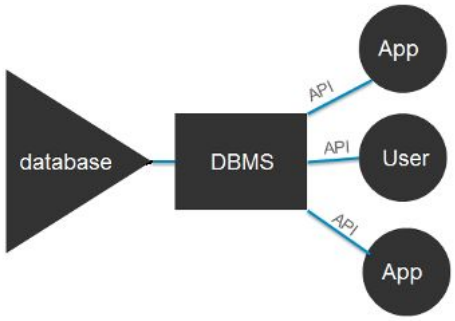
معمولا بازی های آنلاین توسط یک کلاینت به سرورهای خودشان متصل می شوند. روش‌های این اتصال می توانند متفاوت باشد. در بعضی بازی‌ها می توان سرور را انتخاب کرد اما در اینجا شما تنها نیاز دارید که از یک سرور استفاده کنید. کلاینت شما تمام گرافیک و انیمیشن های بازی را مدیریت می کند. اما اجرا کردن این دستورالعمل‌ها منوط به دریافت پاسخ از سرور خود است. دقت کنید که صفحاتی مثل منوها، تنظیمات و ... سمت کلاینت هستند و منطق مربوط به آنها باید در کلاینت شما پیاده شود. هر جا که نیاز بود نیز کلاینت با سرور خود ارتباط برقرار می کند. کلاینت شما نیز fault tolerant است یعنی در صورت بروز خطا به هر علتی می تواند به درستی آن را مدیریت کند.

در نهایت توجه کنید در این بخش بازی شما باید توانایی بازی online و offline را به کاربران بدهد. سعی کنید راه حلی ارئه دهید که برنامه بتواند هر دو حالت را به راحتی پیاده کند بدون اینکه نیاز به تکرار کد و... باشد.

بخش پنجم - قسمت اول - DataBase

تا به اینجا تمام اطلاعات بازی شما توسط فایل‌ها مدیریت و ذخیره می شدند. در این بخش پروژه ، ذخیره سازی این اطلاعات روی دیتابیس مدنظر است.

دیتابیس ها در واقع سیستمی کامل برای مدیریت داده‌ها هستند. در واقع DBMS (Database Management Systems) ما را از درگیر شدن با ساختمان های داده پیچیده و خود ساختار دیتابیس راحت می کند.

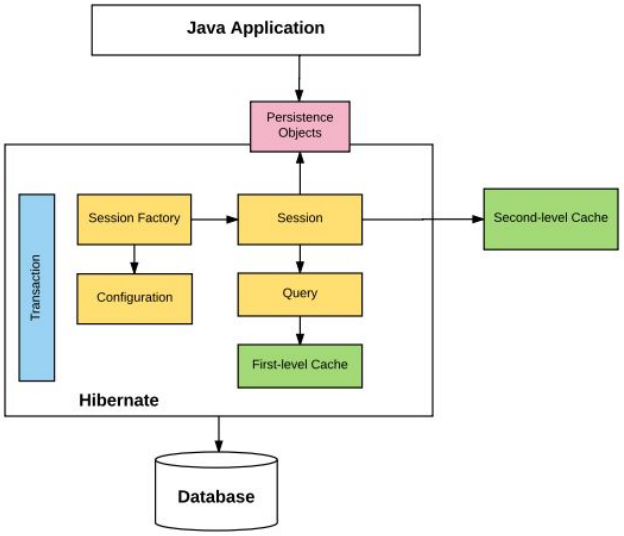


تعدادی از سیستم های دیتابیس:

- MySQL
- PostgreSQL
- NoSQL
- Oracle Database

هر کدام از این سیستم ها قابلیت خاص خود را دارند، مثلا سیستم های NoSQL حجم بزرگ داده ها را مدیریت می کنند و ...

اما در واقعیت، بسیاری از شرکت های فناوری برای مدیریت سیستم های خود از ORM ها استفاده می کنند. این سیستم ها به توسعه دهندگان(developers) اجازه می دهند که بسیار راحت تر با داده‌ها ارتباط برقرار کنند. در این بخش شما از مشهورترین ORM در جاوا یعنی hibernate استفاده خواهید کرد.



شما با استفاده از دستور های هایبرنت باید داده های زیر را در دیتابیس خود ذخیره کنید:

- لاگ‌های بازی سمت سرور
- اطلاعات کارت‌ها
- اطلاعات قهرمان‌ها
- اطلاعات بازیکن‌ها

.

پیاده سازی درست بخش های قبلی مخصوصا بخش 1 در اینجا می تواند به شدت به شما کمک کند. در بخش یک شما باید داده‌ها و مدل های خود را درست ذخیره و نگهداری می کردید. اگر این کار را خوب انجام داده باشید می توانید به راحتی این داده‌های را از روی فایل به روی دیتابیس انتقال بدهید.

بجز پیاده سازی اجزای گفته شده، پیاده سازی تمیز و اصولی بخش دیتابیس(می توانید از داکیومنت‌های هایبرنیت استفاده کنید) در ارزیابی شما تاثیر گذار است. سعی کنید با استفاده از مالتی تردینگ و برنامه نویسی موازی وظایف دیتابیس را جدا کنید. همچنین اقدامات لازم در صورت قطع اتصال با دیتابیس را به سرور خودتان اضافه کنید. (این که ارورها چگونه مدیریت شود را به عهده خلاقیت خودتان می گذاریم!)

بخش پنجم - قسمت دوم - Reflection

از ویژگی های پیشرفته جاوا این است که می توانیم بعد از کامپایل شدن و اجرا شدن برنامه، به آن ویژگی های جدید اضافه کنیم. در این بخش نهایی پروژه، از شما میخواهیم که این سیستم را نیز اضافه کنید.

فرض کنید که سرور بازی شما واقعاً در حال اجرا شدن در یک سرور فیزیکی است و بازیکنان زیادی به آن متصل اند. بازی شما برای پیشرفت و توسعه نیاز به دریافت آپدیت‌های متعددی دارد. هرچقدر که آپدیت کردن کلاینت‌ها آسان است(می توانید نسخه جدید بازی را برای آنها ارسال کنید)، آپدیت کردن سرور مشکل است. در واقع نیاز است که سرور را غیر فعال کنید و سپس نسخه جدید را بارگذاری کنید و ...

اگر آپدیت‌های شما کوچک باشد (اضافه کردن یک کارت،قهرمان و ...) بهتر است که از ویژگی های مثل reflection استفاده کنیم.

ادمین سرور شما باید بتواند فایل های کامپایل شده ویژگی‌های جدید را بارگذاری کند و سرور بدون اشکال این فایل های کامپایل شده را در زمان اجرا در بازی اعمال کند. جزئیات دقیق این رویداد در آینده به شما گفته می شود.

- بار دیگر اهمیت رعایت اصول برنامه نویسی شی گرا و برنامه نویسی تمیز در اینجا خود را نشان می دهد. بدیهی است که اگر طراحی درستی برای کلاس های بازی خود نداشته باشید نمی توانید به راحتی ویژگی جدیدی اضافه کنید.

موفق و پیروز باشید

تیم درس برنامه نویسی پیشرفته