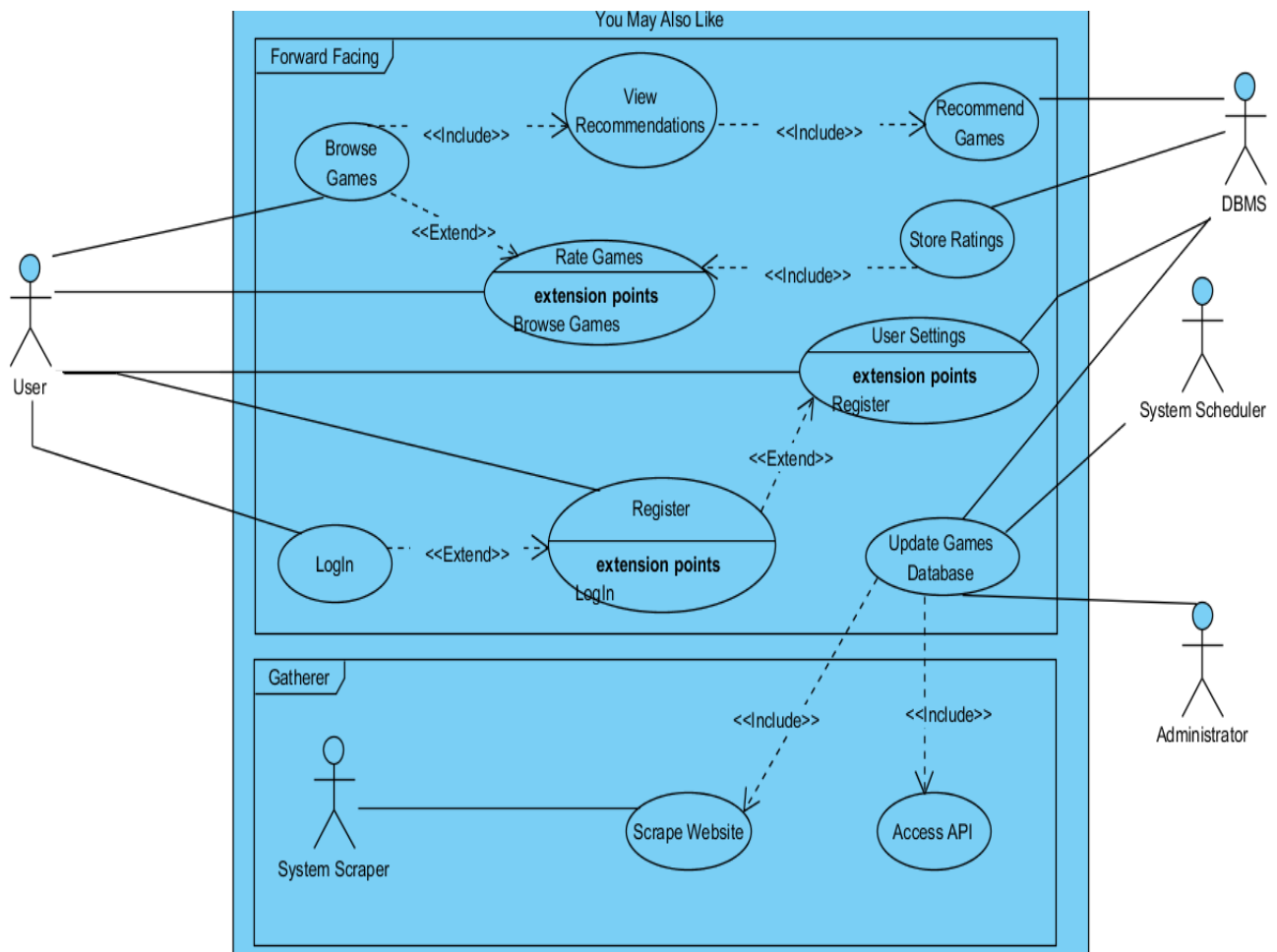


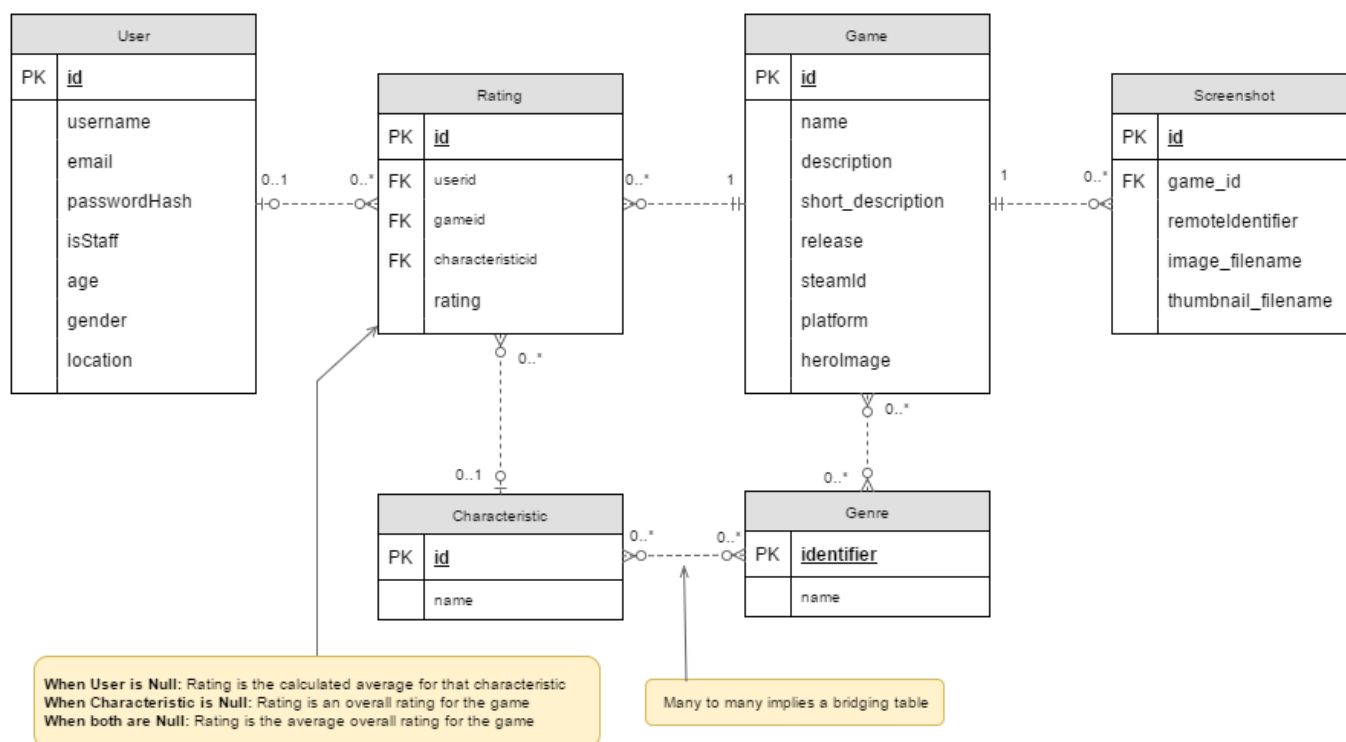
Requirement Model

“You Might Also Like...”

Use-Case Model



Database Design



Use-Case Descriptions

1.1. Browse Games

When authenticated User

Wants to view the games within the system they visit the “browse” page

So that they can see the variety of games that are registered with the system or can see details on a specific game including description, current ratings, posts from other gamers in regards to their feedback to the game, can also rate a game. Further implementation could include ability to purchase a game. This page also lists trending games based on analytics on gamers gaming behaviour and its match with other users of similar gaming behaviour rating the games they like.

Includes use-case 1.3. (View Recommendations)

1.2. Rate Games

When authenticated User

Wants to rate a game within the system they can simply do so on the browse page itself

So that they can provide specific ratings to the games they desire and can also post their speak in relation to the game as feedback

Extends use-case 1.1. (Browse Games)

1.3. View Recommendations

When authenticated User

Wants to view the current list of games recommended for them they visit the home page or “recommended” page

So that the User can view all games recommended by the system based on their preferences

Is included by use-case 1.1. (Browse Games)

Includes use case 1.4. (Recommend Games)

1.4. Recommend Games

When authenticated user in on Home Page a.k.a. browse page DBMS simply displays recommended games

Wants to view the current list of games recommended for them they visit the home page or “recommended” page

So that the User can view all games recommended by the system based on their preferences

Is included by use-case 1.3. (View Recommendations)

1.5. Store Ratings

When authenticated user rates a game, DBMS updates and stores data on rating provided

Wants to provide feedback on a specific game via a rating system

So that the User can provide feedback and DBMS can store data for further analytics

Is included by use-case 1.2. (Rate Games)

1.6. Login

When User

Wants to login to the system they click login and enter their account details

So that the User is authenticated and can perform actions that require authentication

1.7. Register

When User

Wants to register an account for the service they click register, and fill out the requested information

So that the User is granted an account and the system has the required information about them

Extends use-case 1.6. (Login)

1.8. User Settings

When authenticated User

Wants to update their settings/account information they visit the settings page and alter any information/settings

they wish, they can also specify the game preferences under this which can be later updated as and when required

So that the User’s settings/info is updated and the system reflects this

Extends use-case 1.7. (Register)

*** Notes - DBMS is subsequently updated with data on login, register and User settings as depicted by the Use Case diagram

2.1. Update Games Database

When the Scheduler

Triggers based on settings it begins the process of updating the games database

OR

When an Administrator

Wants to manually trigger the process of updating the games database they visit the administrator page and trigger the update

So that the system will begin launching scrapers and accessing APIs to grab game lists from external sources

Includes use-cases 2.2. And 2.3. (Scrape Website and Access API)

2.2. Scrape Website

When the System

Launches the Scraper the Scraper searches the entirety of the website for relevant data (game info)

So that the system’s database can be updated despite the target data not having a public API

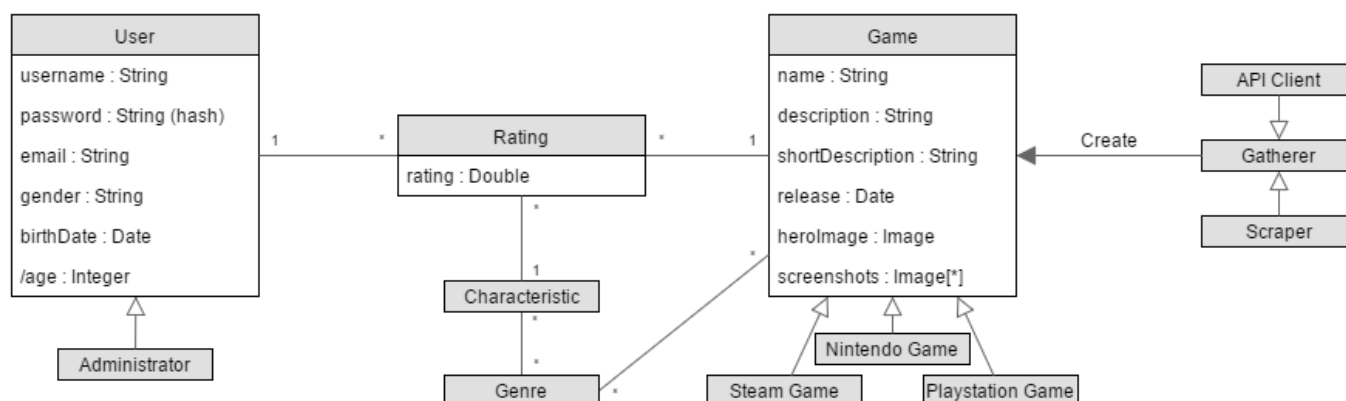
2.3. Access API

When the System

Wants to update its database using a service that provides a public API they run a series of API calls

So that the system’s database can be updated with the new data

Domain Model



Non-Functional Requirements

Can be found in the separate NFR list on version control and below:

You May Also Like / You May Dig a.k.a. YMD

NFR	Trigger Question	Impact	Answer	Strategy	Priority
Security	<p>Will the login access be created via social media accounts such as Facebook OR email services such as Gmail or the log in will be platform/software independent i.e. will this software be able to plug in /integrate to any other platform/software such as games , Netflix, or any other service or choice offering platform and will use the existing login details that is already there as the user will probably be already logged in</p> <p>Will there be any payment required to obtain any premium features assuming that there will be a free version available</p>	<p>Loss of business/customer data,</p> <p>Abuse of payment details by unauthorized third parties,</p> <p>Loss of customer's confidence and eventually revenue loss</p>	<p>Authorized access only</p> <p>Payment card industry compliant</p> <p>Transaction and activity logs recorded</p>	<p>Terms and conditions provided to the user at the time of registration or payment authorization</p> <p>Clarity on what information will be accessed of which account including payment</p> <p>Re login required upon inactivity of say two minutes</p>	Medium
Audit	Will usage behavior be recorded, analyzed and audited with a time stamp	Recommendations provided may not generate any matching interests for the user	All transactions and activity recorded and analyzed in timely manner	Smart use of analytics via cognitive learning	High
Performance	Will the recommendation	Irritated and frustrated user, Confused users,	Algorithm needs to be effective to determine the	Low bandwidth	High

	<p>provided be relevant</p> <p>Will the platform be able detect ever changing usage behavior and able to adapt</p> <p>Will browsing through the recommendation list of items be simple offering easy navigation and have a logical reason behind the recommendation</p>	<p>Loss of consumers due to unpleasant experience while browsing</p> <p>Irrelevant recommendations will generate no interest from users making this an useless exercise</p>	<p>user interest and needs to be cognitive to adapt t evert changing usage behavior or if a user has wide variety of taste...then the recommendations needs to be relevant enough</p>	<p>requirements, Machine (computer or mobile) independent performance, easy navigation via low page load requirements</p>	
Capacity	<p>Will the traffic of users be supported at all times</p>	<p>Many customers unable to use the web system as and when required/desired. Loss of revenue due to users not returning back to use the system due to frustrating experience</p>	<p>Enough hosting bandwidth allocated, infrastructure should be able to support unlimited concurrent user via elastic server and storage environment</p>	<p>Elastic infrastructure deployment including server, bandwidth and data center arrangement</p>	Low
Availability	<p>Will the uptime of web system be 100%</p>	<p>Loss of confidence among users including raised concerns about the privacy of their data if the system is down</p>	<p>99.999% uptime guaranteed infrastructure and backend support No downtimes and quick recovery plans</p>	<p>Legal vendor agreements in place for infrastructure with strict SLA, Penalties in the instance of breach , Incentive plans for infrastructure managers for 100%</p>	Medium

				uptime in timely manner, Backup links and backup to backup links in place for internet and hardware/software components	
Reliability	Will the system be available and functional at all times?	Frequent failure of the system could result in loss of consumers and revenue of business	The mean time between failure rates should be minimal. If the system does fail the recovery should be instant	Legal vendor agreements in place for infrastructure with strict SLA, Penalties in the instance of breach , Incentive plans for infrastructure managers for 100% uptime in timely manner, Backup links and backup to backup links in place for internet and hardware/software components	Low

Recovery	Will the down time be minimal with no usage impact to the consumers	The increased downtime could result in frustration of consumer's faith in the company as it is a common practice to be recovery instantly fixed. This could result in raised data security concerns among users as well again reducing revenue and number of users	The recovery should be instant within a split second. It is easy to deploy solutions like these and again is a common practice	Legal vendor agreements in place for infrastructure with strict SLA, Penalties in the instance of breach , Incentive plans for infrastructure managers for 100% uptime in timely manner, Backup links and backup to backup links in place for internet and hardware/ software components	Low
Compatibility	How will the system work on different platforms such as iOS, android, windows and web. Will we need to integrate with payment card companies?	Nonfunctional system with transaction errors and increased time to complete transactions, frequent crashes on different platforms. Frustrating user experience	This should be a key part of testing process before any production goes live and is available for the user to experience.	Testing procedure in place for any integration or deployment of any upgrades or error fix.	High
Maintainability	Will there be any frequent updates/upgrades required	Increased downtime and increased frustration among users	The maintenance work should be done in the background and should have no	The regular maintenance should	Medium

			impact on the up time of the system	have zero impact on the up time. This could again be part of vendor and infrastructure manager SLA. If the downtime is unavoidable then the users should be notified beforehand.	
Usability	Will the system be easy to use	Complex and non-user friendly platforms are never the popular ones and normally are often replaced by a competitive system concentrated merely on user friendliness. Failure to deploy a user friendly system could result in no business at all often coined as dinosaurs of systems as could go extinct easily	The system should cater to people of all ages and computer literacy, should be pleasant to use and should be consistent across all pages of its existence. The error prevention techniques should be used to avoid transaction errors.	This is a very complex and a broad concept. It derives from several fields including but not limited to heuristic evaluations, psychology, expert panels, graphics and animations. Significant time should be spent on this	Extremely High

				concept including its R & D. The system should be able to cater to people of all ages including , all sorts of language and computer literacy skills	
Documentation	Will users have help docs available including their legal rights entitlement	Lack of privacy policies and legal entitlements declaration could be a legal breach with serious consequences. Help docs and FAQ could enhance user experience	Documentations tab available on all pages layout available for the user to click on to go through a list of available uploaded docs and links	Regular uploads at the backend of this doc tab and available links .constant updates to the doc if required to keep it updated	Low
Integrity	Will we have a log of data validation and fault tracking	Less visibility could pile on errors over time and reduce security and would make it impossible to enhance user experience if no analytics could be run on this log	This is a complex thing to deploy and should be carefully considered .Would require a lot of business use cases	Some use cases could be deployed in the start of the system development. This however needs to be part of the system maintenance and needs to be a	Low

				cognitive practice i.e. learn and deploy.	
--	--	--	--	---	--