Gotherator
System Requirements

0. Document Details

0.1 About This Document

This document represents the complete system requirements for the Gotherator System. This is not a software design document, but rather details *what* the system-to-be is. It is loosely based off of the V*olere* template.

0.2 Change log

note: Iteration matches the currently working release/tag

Iteration / Date	Section Changed / Added	Brief Description
V0.0.0 / 12/31/2017	Initial Commit	Initial Commit

0.3 Semantic Versioning

- Releases prior to v1.0.0 (0.X.Y)
  - X denotes a milestone, as listed on Github. Each milestone will contain an agile iteration of partial or complete implementation of new features/functions/requirements.
  - Y denotes notable changes between each milestone, including:
    - Documentation/Requirements updates
    - Bug Fixes – Although versions prior to v1.0.0 are incomplete, versions of the master branch should be functional and bug-free

0.4 Work Schedule

This is a side-project. Thus the following schedule shall be used for iteration of features and requirements:

- Every Saturday-Sunday is documentation review.
  - The SRS is intended to change very minimally over the course of the project.
  - Design documentation is reviewed. These include UI mocks and software module design
  - Milestone and Roadmap planning is reviewed on Github
- The current work schedule is as follows
  - Saturday-Sunday: at liberty
  - Wednesday: main work day

1. Project Drivers

1.1 Project Purpose

1.1.a) Gotherator BusinessStatement
This application seeks to facilitate a safe, anonymous ecosystem for match-based chat.

1.1.b) Goals of the Project

1. To implement modern technologies
2. To integrate with current popular social media platforms
3. To create an engaging, visually appealing chat-based system

1.2 The Client, The Customer, and Other Stakeholders

1.2.a),b) The Client & Customer
The developer(s) of the project are the current clients.

1.2.c) Other Stakeholders

- Facebook (via API)
- Google (via API)
- OKCupid (via Model)

1.3. Users of the Product

- Facebook Users
- Google+ Users
- OKCupid Users
- Anyone who enjoys anonymous chat

2. Project Constraints

2.1 Mandated Constraints

2.1.a) Solution Constraints

- 2.1.a.1 – The product shall contain a protected API
- 2.1.a.2 – The product shall use ReactJS and Redux
- 2.1.a.3 – The product shall be developed using Websockets

2.1.b) Implementation Environment of the Current System

- Recent versions of modern web and mobile browsers

2.1.c),d) Partner or Collaborative Applications and off-the-shelf Software

- Facebook Applications **(collaborate)**
- Google (**collaborate)**
- Express, Sequelize, ReactJS, Redux, **(off-the-shelf)**

2.2 Naming Conventions and Definitions

Term/Acronym	Description/Definition
Code of Conduct	A set of rules and guidelines denoting a contract for how a user should use the application
Display Name	A unique name/handle/string visible to other users
Display Themes	Visual styles in the application. May include background art, colours, typography and more.
Github	Remote repository service. See http://www.github.com
Gotherator	The System-To-Be, loosely defined as ‘An application for connecting people through macabre interests’
Issue	Any of the following units: -A bug to be fixed -An inquiry about possible features -A new feature being developed See https://guides.github.com/features/issues/
Likeness	A value denoting how compatible users
Match Criteria	Various forms of media (such as binary questions, images) that require a user to unambiguously respond in some manner
Milestone	A group of issues that belong to a unit such as a project, feature, version iteration See https://guides.github.com/features/issues/#filtering
Moderator	A user with extra privileges compared to regular users. These privileges allow these users to filter media in the application, thus effecting the application’s content.
Project	A roadmap for a major SemVar releaes. May contain various Issues, Milestones and Features. See https://help.github.com/articles/about-project-boards/
SemVer	Semantic Versioning See https://semver.org

2.3 Relevant Facts and Assumptions

2.3.a) Facts

- Many people feel that dating services are unsafe or unforgiving to predators
- Many people wish to connect with those that have similar interests

2.3.b) Assumptions

- Users will be less likely to abuse other users if they have no control over search criteria and interactions are kept anonymous
- User-moderation/reporting will keep predators away

3. Scope and Context

3.1 Scope and Context

3.1.a) Scope

The context diagram in **section 3.1.b** shows the system context. However, it is also useful to set out aspects that are explicitly out of scope in order to manage scope creep.

- The system will not permanently store chat histories

3.1.b) Context Diagram

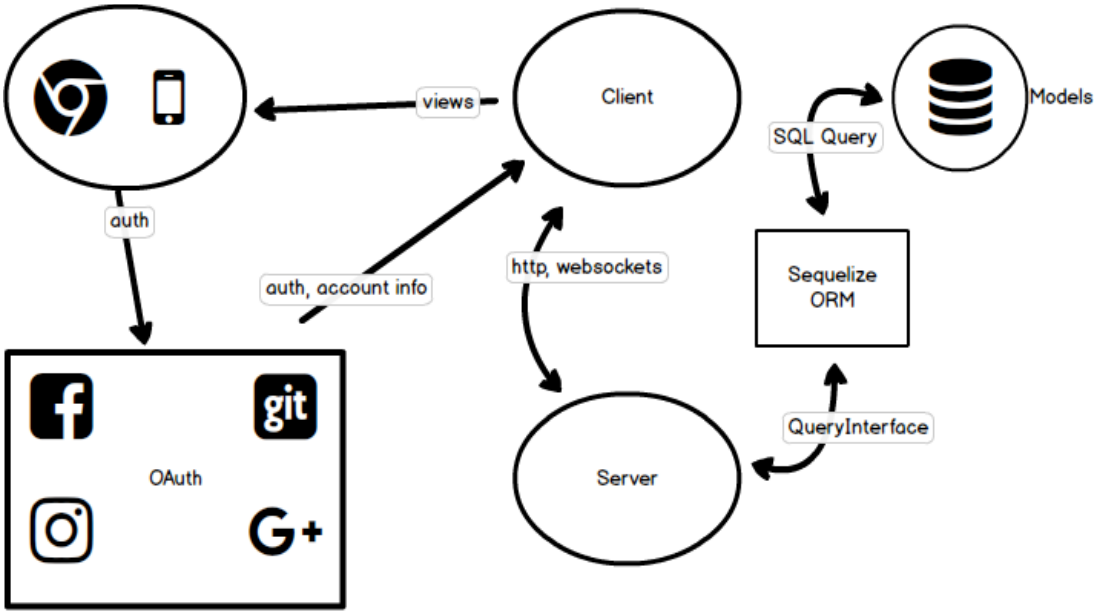

Illustration 1: Context Diagram

3.1.c) Use-Cases

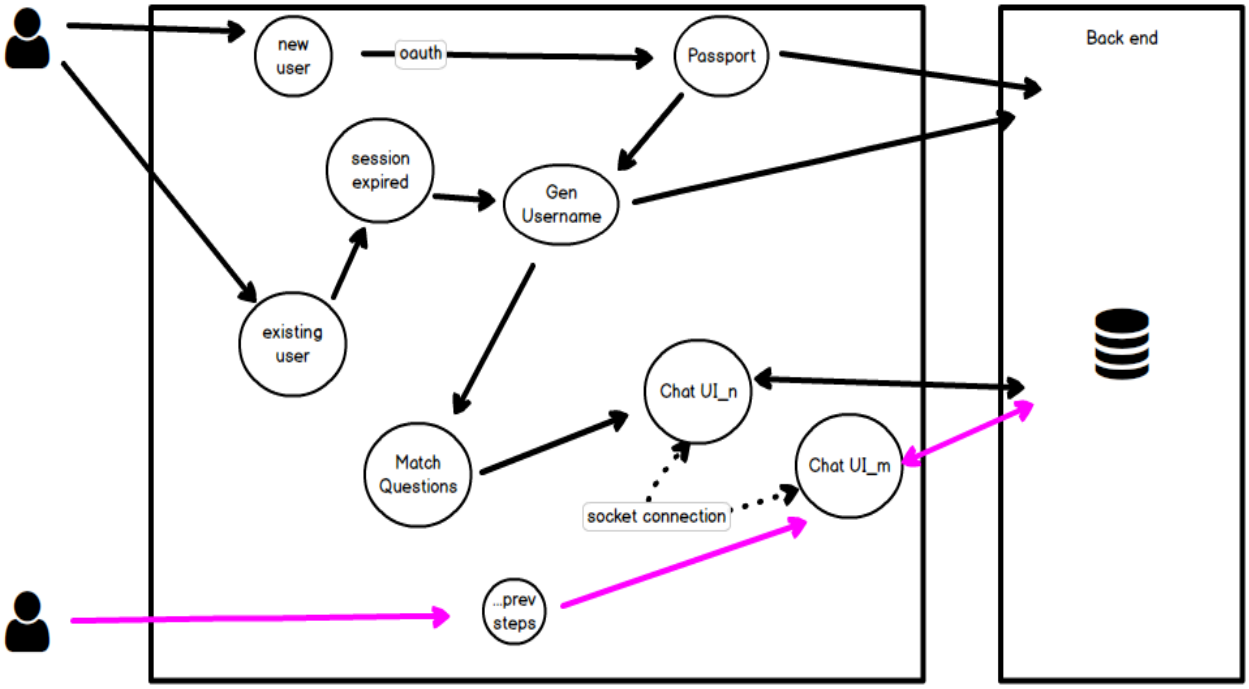

Illustration 2: User Use-Cases

4. Functional and Data Requirements

4.1 Functional Requirements

- 4.1.1 – The product shall authenticate users through other social media accounts
- 4.1.2 – The product shall generate display names for users
- 4.1.3 – The product shall record each user's social media profile(s)
- 4.1.4 – The product shall record each user's answers to match criteria
- 4.1.5 – The product shall calculate each user's likeness to one another
- 4.1.6 – The product shall prompt users to enter private chat with matched users
- 4.1.7 – The product shall allow users to reveal their identity to one another
- 4.1.8 – The product shall record other users one has chatted with prior
- 4.1.9 – The product shall change each user's display name upon successive uses
- 4.1.10 – The product shall allow users to report one another for behaviour that breaches the code of conduct
- 4.1.11 – The product shall have a code of conduct that each user can view
- 4.1.12 – The product shall allow users in chat to prompt one another with match criteria
- 4.1.13 – The product shall allow users to change display themes
- 4.1.14 – The product shall have chat rooms for multiple users to chat
- 4.1.15 – The product shall allow users to submit artwork to appear in the application
- 4.1.16 – The product shall allow users to create match criteria to appear in the application
- 4.1.17 – The product shall upgrade specific users with extra privileges, called moderators
- 4.1.18 – Moderators of the product shall be granted privilege to vote on modifications to existing match criteria
- 4.1.19 – Moderators of the product shall vote be granted privilege to vote on user submissions

4.2 – Data Requirements

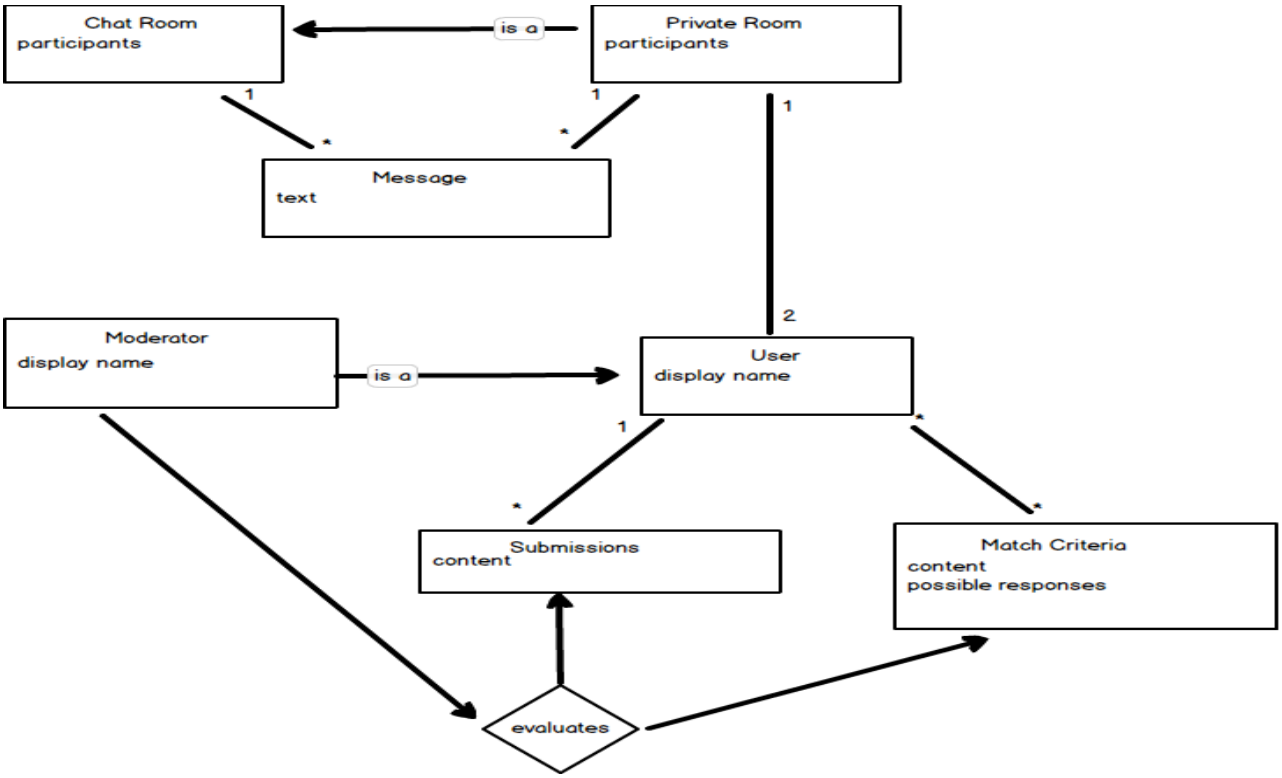

Illustration 3: Domain Model

4.2.a) User > Moderator

The User object represents an authorized Gotherator account. This object is used to map match criteria, chat room membership and submissions to a particular user:

- Records Submissions a User has created to their account (User Has Many Submissions)
- Records Match Criteria to a User (User has many Match Criteria)
- Moderators status must be recorded

4.1.b) Chat Room > Private Room

The Chat Room object represents a set of Users and Messages exchanged between them.

- Public Chat Rooms (Have Many Users, Have Many Messages)
- Private Chat Rooms (Have 2 Users, Have Many Messages)

4.1.c) Match Criteria

The Match Criteria object maps Users to Responses

- Records which users have responded (Has Many Users)
- Contains media content (Such as a question or image)

5. Non-Functional Requirements

5.1 Look And Feel

Many of these requirements shall be documented with mockups/wireframes. There are still some general requirements:

5.1.1 Appearance Requirements

- 5.1.1.a) The product shall be attractive to counter-cultures
- 5.1.1.b) The product shall appear community-driven
- 5.1.1.c) The product shall not imply any corporate branding

5.1.2 Style Requirements

- 5.1.2.a) The product shall have a macabre spirit
- 5.1.2.b) The product shall use dark colours
- 5.1.3.c) The product shall use gothic typography

5.2 Usability And Humanity Requirements

5.2.1 Ease of Use Requirements

- 5.2.1.a) The product should be easy for 16-year-olds and up to use
- 5.2.1.b) The product should be used by people with no training and basic understanding of English
- 5.2.1.c) The product interface should imply use
- 5.2.1.d) The product shouldn’t require any memorization for successive uses

5.2.2 Personalization and Internationalization Requirements

- 5.2.2.a) The product should allow the user to select a chosen language
- 5.2.2.b) The product should allow the user preferred display styles/themes

5.2.3 Accessibility Requirements

- 5.2.3.a) The product should attempt to be accessible by partially sighted users
- 5.2.3.b) The product should be fully usable by users with auditory impairments

5.3 Performance Requirements

The system-to-be is being designed primarily for use in North-America by an overall number of users on the order of magnitude 10^4. Still, scaling and performance are accounted for via DevOps and deployment options. Theoretically, scaling/regional clustering should be possible with the design.

5.3.1 Speed and Latency Requirements

- 5.3.1.a) The product should have a response time of < 400ms **(Doherty Threshold)**
- 5.3.1.b) The product should facilitate Real-Time chat

5.3.2 Reliability and Availability Requirements

- 5.3.2.a) The product shall be available 24/7

5.3.3 Capacity Requirements

- 5.3.3.a) The product shall be able to cater to 1000s of simultaneous users

5.3.4 Scalability Requirements

- 5.3.4.a) The product shall be horizontally scalable

5.4 Operational and Environmental Requirements

The expected physical environment is user's web and mobile browsers. However, vendor APIs will be used which means that the product will have to maintain/conform to their API changes.

5.4.1 Requirements for Interfacing with Adjacent Systems

- 5.4.1.a) The product should work on the last several releases of the most popular web and mobile browsers
- 5.4.1.b) The product should work with the current authorization protocols/APIs
  - Oauth2.0
  - Facebook, Google etc.

5.4.2 Productization Requirements

- 5.4.2.a) The product should be distributed as web URI
- 5.4.2.b) The product should be able to be distributed as a mobile native application in the future
- 5.4.2.c) The product should be distributed as a ZIP file for open source contribution

5.4.3 Release Requirements

Refer to **Section 0.3 Semantic Versioning**

- 5.4.3.a) The product should contain a link to its Github Repository for bug reporting

6.5 Maintainability and Support Requirements

- 6.5.1 The product should be maintained on a weekly basis
- 6.5.2 The product should not require a high level of support. The developer's Github repository and email will remain public for discussion, questions and misconduct reports.
- 6.5.3 Should the product scale to requiring more support, misconduct will be handled by Moderators.

6.6 Security Requirements

In general, the product should aim to have as few security vulnerabilities as possible. This is a hobby/community-drive project.

6.6.1 Access Requirements

- 6.6.1.a) Only Authorized Maintainers of the application should have access to database records.
- 6.6.1.b) Only Users with Moderator status should have access to pending submissions and evaluating existing match criteria

6.6.2 Integrity Requirements

- 6.6.2.a) The product should attempt to prevent incorrect data from being introduced
- 6.6.3.b) The product should not facilitate any commerce

6.6.3 Privacy Requirements

- 6.6.3.a) The product should make its users aware of its information practices before collecting data from them
- 6.6.3.b) The product should notify users of changes to its information policy
- 6.6.3.c) The product should not reveal private information to any third parties, advertisers etc. The exception is the developers and the cloud hosting platform whereby the data resides.

6.7 Cultural and Political Requirements

In general, the product should aim to be a safe, enjoyable community for counter-cultures.

6.7.1 Cultural Requirements

- 6.7.1.a) The product shall remove users that discriminate on a basis of culture, religion, ethnicity, sexual orientation, gender.

6.7.2 Political Requirements

- 6.7.2.a) The product shall align itself with anti-fascism, anti-nazism and related ideologies.

6. Project Issues

6.1 Open Issues

Open Issues shall be documented on the Github Repository. To differentiate them from bugs and feature requests, they will be tagged with a label of **question**

6.2 Off-The-Shelf Solutions

The product shall use many off-the-shelf solutions. It is impractical to list all given that various NPM modules will be used. Critical solutions include:

1. PostgresSQL database with Sequelize ORM
2. PassportJS Authentication
    1. Facebook OAuth 2.0 Strategy
    2. Google OAuth 2.0 Strategy
3. Node and Express JS
4. Websockets via websockets/ws Node module
5. ReactJS, Redux via React-Redux
6. React-Router-Dom

6.3 Tasks

Refer to **sections 0.1-0.4** of this document for a description of task management and work scheduling.

6.4 Risks

- Inaccurate Match Criteria Metrics
- Underestimating Scale/Network Requests
    - This may evoke a surcharge that the developer is not prepared to encounter
- Low Quality

6.5 Costs

- The project should try to incur **ZERO** costs during/after development
- If the product takes off, crowd-sourcing and support shall be requested through platforms such as Patreon
- Under no circumstances shall users of the product be required or prompted for payment