SOFTWARE REQUIREMENTS SPECIFICATION [V3]

Software Requirements Specification for a Game Streaming Application called G Stream

OCTOBER 27, 2023

BY HD SECTOR

Contents

1. Introduction 3
   1.1 Purpose 3
   1.2 Intended Use 3
   1.3 Scope 3
2. Overall Description 3
   2.1. Product Perspective 3
   2.2. Product Features 4
   2.3. User Classes 4
   2.4. Operating Environment 4
3. Specific Requirements 4
   3.1. Functional Requirements 4
      3.1.1. User Registration 4
      3.1.2. User Authentication 4
      3.1.3. Profile Management 5
      3.1.4. Content Discovery 5
      3.1.5. Live Streaming 5
      3.1.6. Chat and Interaction 5
      3.1.7. Notifications 5
      3.1.8. Monetization 5
      3.1.9. Content Reporting and Moderation 5
   3.2. Non-Functional Requirements 5
      3.2.1. Performance 5
      3.2.2. Usability: 6
      3.2.3. Maintainability 6
      3.2.4. Flexibility 6
      3.2.5. Security 6
      3.2.6. Compatibility 6
      3.2.7. Data Privacy 6
   3.3 Technology Requirements 6
      3.3.1. Programming Language 6
      3.3.2. Data Requirements 7
4. Methodology Required 7
   4.1. SDLC Models 7

Waterfall Model 8
RAD Model 8
Spiral Model 8
V-Model 8
Incremental Model 9
Agile Model 9
Iterative Model 9
Big bang model 9
Prototype Model 9
4.2. The suitable SDLC Model for the specific Application Development 10
4.3. Phases of Agile Model 10
4.3.1 Requirements: 10
4.3.2 Design: 11
4.3.3 Development /Iteration: 11
4.3.4 Testing: 11
4.3.5 Deployment: 11
4.3.6 Review 11
4.4. Why choose Agile Model 12
5. System Models/Diagram 13
USE CASE DIAGRAM 14
Flow Chart 15
UML DIAGRAM 16
6. Test and Validation 17
7. Project Timeline and Gantt Chart 17
8. Payment Terms 18
9. Contact 18

1. Introduction

1.1 Purpose

The main purpose of the G Stream Streaming App is to provide gamers with a robust platform to stream their gameplay live and to offer viewers an engaging way to enjoy and interact with gaming content in real time. The app aims to foster a dynamic gaming community and enhance the overall gaming experience for both streamers and viewers.

1.2 Intended Use

The Game Streaming App enables gamers to broadcast their gameplay sessions in real time, allowing viewers to watch and interact with the streamers. It facilitates engagement through live chat, comments, and reactions, creating a social and interactive experience around gaming content.

1.3 Scope

The Game Streaming App will encompass the following features:

- The User registration and authentication process to ensure secure access.
- User profiles for streamers and viewers with customizable avatar and profile information.
- Live streaming functionality that allows streamers to broadcast their gameplay sessions with live audience.
- Interactive features like live chat, comments, and real-time reactions and donations.
- Search and discovery tools to find streams based on game titles, genres, and streamers profiles.
- Notifications for the followers and subscribers about new streams and interactions.
- Analytics dashboard for streamers to track viewership, engagement, and trends in gaming community.
- Cross-platform compatibility for major operating systems (iOS, Android, Windows & macOS).

2. Overall Description

2.1. Product Perspective

The G Stream app operates as a standalone live streaming platform, enabling users to discover, watch, and interact with live broadcasts from streamers. It is part of the larger G Stream ecosystem, which includes a web platform and APIs for third-party integrations.

2.2. Product Features

Key features of the G Stream game streaming app include:
- User registration and authentication.
- User profile and information management.
- Content discovery through categories, tags and recommendations.
- Live streaming with live chat functionality.
- Real-time notifications for followers and their favorite streamers.
- Monetization options for streamers.
- Content reporting and moderation tools.

2.3. User Classes

- Viewers: The Individuals who are interested in watching live streams.
- Streamers: Content creators or streamers who broadcast live.
- Moderators: Trusted users or admins responsible for maintaining platform guidelines.

2.4. Operating Environment

The app is designed to run on iOS (above IOS 7) and Android smartphones (version 6.0 and above) and different web browsers (supporting modern standards).

3. Specific Requirements

3.1. Functional Requirements

3.1.1. User Registration

- Users can sign up using email, phone number, or social media accounts (e.g. twitter, Facebook).
- Validation of user-provided information during registration of profile.
- Account verification via email or SMS by code.

3.1.2. User Authentication

- Secure and safe login mechanism with password encryption.
- Support for two-factor authentication (2FA).
- Session management for the logged-in users.

3.1.3. Profile Management

- Users can create and edit their profiles with their own information.
- Profile includes display name, profile picture, and brief description of user.
- Option to link social media profiles.

3.1.4. Content Discovery

- Browse live streams by category, tags, and popularity of games and streamers.
- Personalized content recommendations based on user preferences and viewing history in profile.

3.1.5. Live Streaming

- Streamers can start and watch live streaming of other players.
- Viewers can watch live streams and interact through chat simultaneously.

3.1.6. Chat and Interaction

- Real-time chat for viewers to interact with streamers and other viewers.
- Moderation tools for streamers and moderators to manage chat.
- Emote and gifting system for viewer engagement.

3.1.7. Notifications

- Push notifications for follower activity (e.g., new followers, live stream alerts).
- Option to customize notification preferences.

3.1.8. Monetization

- Streamers can monetize their content through ads, donations, and subscription models.
- Integration with payment gateways (e.g. PayPal, Google Pay) for transactions and donations.
- Revenue tracking and withdrawal options for the streamers.

3.1.9. Content Reporting and Moderation

- Users can report any inappropriate content if they want.
- Moderators can review and take action on reported content and users.
- Automated content flagging for potential violations.

3.2. Non-Functional Requirements

3.2.1. Performance

- App should load quickly and stream smoothly on various network conditions.

3.2.2. Usability:

Usability of software will be easy so that e-learner can use it without any difficulty.

3.2.3. Maintainability

Software would build up in such a way that classifications of errors and maintenance of mechanism become easy.

3.2.4. Flexibility

Software would be flexible so that it can easily accept all changes at low cost, time and experience.

3.2.5. Security

- Secure storage and transmission of user data.
- Regular security audits and vulnerability assessments.

3.2.6. Compatibility

- Support for major web browsers like Chrome, Firefox, Safari and mobile platforms (iOS, Android).
- Responsive design for various screen sizes and resolutions.

3.2.7. Data Privacy

- User data encryption at rest and in transit.
- Ensure transparent privacy policy.

3.3 Technology Requirements

3.3.1. Programming Language

Backend: Choose a backend programming language for server-side development. Common choices include Python (Django, Flask), Node.js (Express.js), Ruby (Ruby on Rails), or Java (Spring Boot).
Frontend: Use HTML, CSS, and JavaScript (React, Angular, or Vue.js) for building the best user interface and client-side functionality.

3.3.2. Data Requirements

- Use of a relational database (e.g., MySQL) for user profiles and interactions.
- Use of NoSQL database (e.g., MongoDB) for real-time chat and notifications.

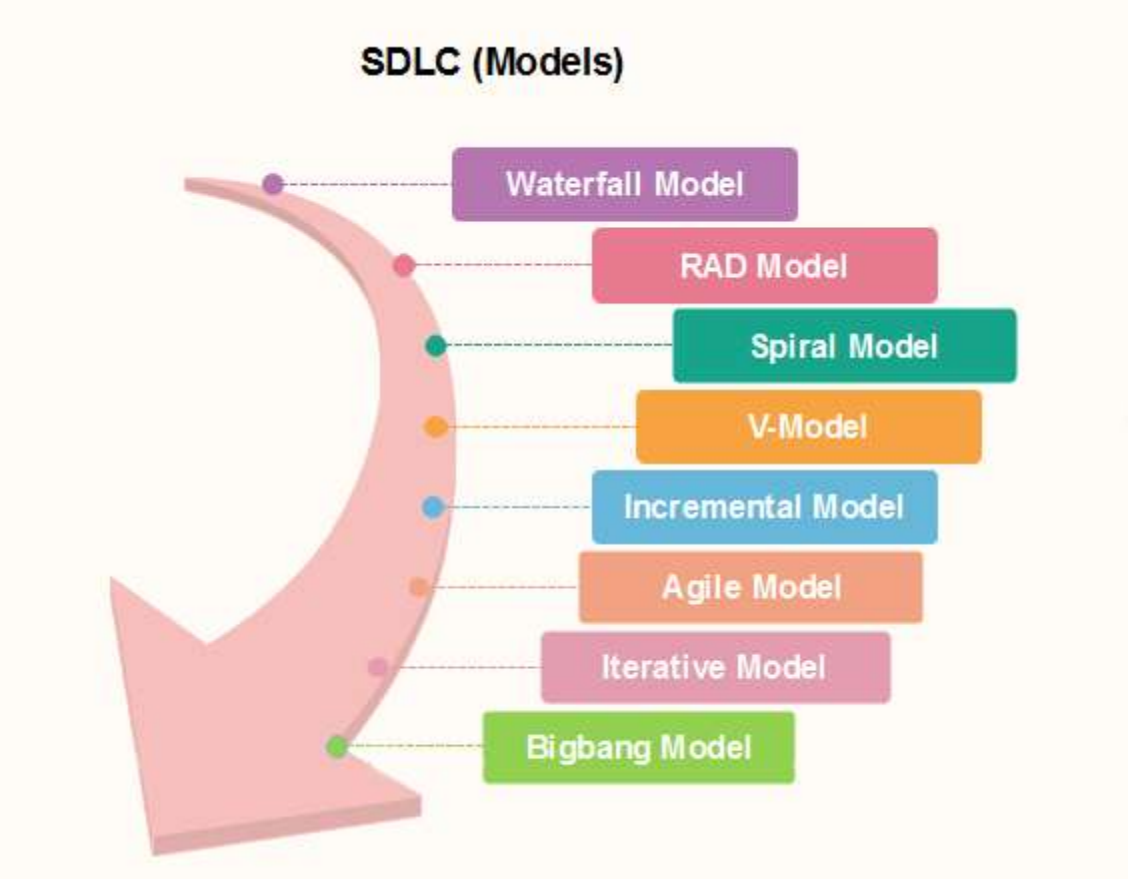
4. Methodology Required

4.1. SDLC Models

Software Development life cycle (SDLC) is a model used in project management that defines the stages include in an information system development project, from an initial feasibility study to the maintenance of the completed application.

There are different software development life cycle models. These models are also called "**Software Development Process Models**." Each SDLC process model follows a series of phase unique to its type for software development.

Here, are some well-known and important models of SDLC life cycle:



Waterfall Model

The waterfall is a universally accepted SDLC model. In this method, the whole process of software development is divided into many phases. The waterfall model is a continuous software development model in which development is seen as flowing steadily downwards like waterfall through the steps of requirements

- analysis
- design
- implementation
- testing integration
- and maintenance.

In waterfall model the project is divided into different phases from upward to downward. In this model each phase is completely wrapped up before the next phase.

RAD Model

RAD or Rapid Application Development process is an adoption of the waterfall model; it targets developing software in a short period of time. The RAD model ensures great system development in lesser time with focused team and implementation.

The phases of RAD model are:

- Data Modelling
- Process Modelling
- Application Generation
- Testing and Turnover

Spiral Model

The spiral model is a risk-driven process model. It is used for risk management that uses elements of one or more process models like a waterfall, incremental etc.

Each cycle in the spiral begins with the identification of objectives for that cycle, the different alternatives that are possible for achieving the goals, and the constraints that exist. This is the first step of the cycle.

The next step is to develop strategies that solve uncertainties and risks. Which involves activities such as benchmarking, simulation, and prototyping.

V-Model

V model also known as Verification and Validation Model. This model works in sequential manner of V shape. Model In this type of SDLC model testing and the development, the step is planned in

parallel. So, there are verification phases on one side and the validation phase on the other side. V-Model joins by Coding phase.

Incremental Model

The incremental model is necessarily a series of waterfall cycles. At the start of the project the requirements are divided into groups. The SDLC process is repeated, with each release adding more functionality also known as “increment” until all requirements are met. The incremental model allows development cycles to overlap. After that subsequent cycle may begin before the previous cycle is complete.

The incremental model can be used for complex and bigger projects.

Agile Model

The Agile Model is combination of incremental and iterative process of software development. It defines each iteration’s number, duration, and scope in advance. In this process the project is divided into multiple iterations and each of them lasts about 3–8 weeks. The division of the entire project into small parts helps to complete the project in less time and with less risk.

Agile Model divides tasks into time boxes to provide specific functionality for the release. Each build is incremental in terms of functionality, with the final build containing all the attributes. Agile Model is perfect for bigger projects and software developments.

Iterative Model

It is a particular implementation of a software development life cycle works based on the initial requirements which are clearly defined and necessary features are added through iterations until the final system is developed.

The process in this model works by multiple iterations that’s why it is called iterative model.

Big bang model

In this model, there are no specific defined process and a very little planning is required. Developers do not follow any specific process. The development starts with the required and necessary funds and efforts in the form of inputs. The customer requirements are also not defined in this model as a result the outcome may or may not be as per the customer's requirement.

Big bang model suitable for smaller and less complex projects.

Prototype Model

The prototype model works by building the prototype first with the requirements gathering, then testing it and again making changes to the prototype until the final and acceptable outcome is

achieved. The developer and the user meet and define the purpose of the software, identify the needs, etc.

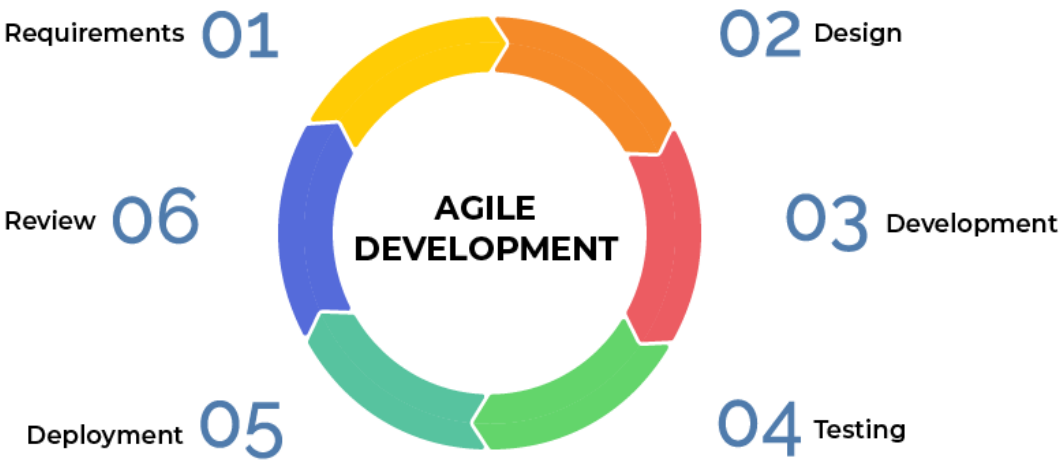
First a quick design or a prototype is created. This design focuses on those aspects of the software that will be visible to the user. It then leads to the development of a prototype. The customer then checks the prototype, and any modifications or changes that are needed are made to the prototype. These prototypes are created using loops and keep getting updates for better outcome.

Once a user is satisfied, the prototype is converted to the actual system with all considerations for quality and security.

4.2. The suitable SDLC Model for the specific Application Development

Agile is based on the adaptive software development methods, whereas the traditional SDLC models like the waterfall model is based on a predictive approach. The Agile Model is combination of incremental and iterative process of software development. In this process the project is divided into multiple iterations and each of them lasts about 3–8 weeks. The division of the entire project into small parts helps minimize delivery time and overall risk.

Users/Customers have the opportunity to make modifications throughout project development phases



4.3. Phases of Agile Model

4.3.1 Requirements:

The requirements or the features are directly taken from the customers or users. Which is written in form of stories or features in collaboration with the stakeholders.

4.3.2 Design:

In Agile, design is an ongoing activity that happens in parallel with development. It focuses on creating just enough design to support the current set of user stories. The designing process in not detailed designing to avoid over-engineering.

4.3.3 Development /Iteration:

The third phase of the agile development process is development or iteration.

In the development or iteration phase, the following steps are accomplished:

- The association with clients.
- Iterations and functionalities are prioritized and implemented.
- Each iteration should be closely examined.
- Delivering regular working software releases.
- Ensuring product quality by testing at regular intervals.

4.3.4 Testing:

The Testing phase in Agile is continuous and integrated throughout the development process. Here automated testing is used to ensure that code changes don't introduce new bugs. For exploratory and usability testing manual testing is also performed.

4.3.5 Deployment:

Continuous Deployment is often a key practice in Agile. It means that code changes are automatically deployed to production as soon as they pass automated tests. This ensures that working software is consistently delivered to users.

4.3.6 Review

Reviews in Agile are part of the sprint cadence. At the end of each iteration/sprint, the team holds a sprint review meeting to demonstrate the completed user stories to stakeholders.
This is an opportunity for stakeholders to provide feedback on the delivered features.

In Agile, these phases are not strictly linear, and they often overlap. The key is to continuously iterate and refine the product. Additionally, Agile encourages adaptability and responsiveness to changing requirements, so adjustments can be made at any point in the process.

4.4. Why choose Agile Model

G Stream is a live streaming platform focused on gaming and esports content. For such an application, a flexible and iterative approach like Agile, with specific attention to Continuous Deployment, would be well-suited.
The reason Agile is best for G Stream application development because Agile development methodology can help reduce the risks associated with complex projects and it ensures that the development team completes the project on given time and with the given budget.

Here's some more reasons:

1. Frequent Feature Delivery: Agile emphasizes delivering small, incremental updates frequently. This aligns with the need to continuously introduce new features, enhancements, and improvements in real-time to keep viewers and streamers engaged on platforms like G Stream.

2. Suitable for Bigger and Complex Project: This is particularly important for large-scale projects, which often involve different priorities and needs. Because Agile methodologies allow teams to act quickly and easily to changes and new requirements.

3. User-Centric Development: Agile model development is mostly a user-centric development. Regular feedback from streamers and viewers can help shape the platform to better meet their needs and preferences, which is crucial for the success of a live streaming service.

4. Flexibility and Adaptability: Agile model is flexible and has great adaptability because allows you to adapt quickly to market shifts and emerging technologies, ensuring that the platform remains competitive.

5. Continuous Improvements: Agile encourages continuous improvement through retrospectives. The development team can reflect on their processes and practices to make ongoing refinements and optimizations.

6. Continuous Integration and Deployment: Agile enable automated testing and deployment, reducing manual errors and speeding up the release process. For a platform that needs to be constantly updated and improved, continuous integration and deployment is critical.

7. Time-to-Market: Agile, along with, Continuous Integration and Deployment allows for faster time-to-market. New features and bug fixes can be deployed quickly and efficiently.

8. Risk Management: Agile ensures early detection of risks through iterative development and continuous testing. This is important for a platform like G Stream, which requires high levels of reliability and stability.

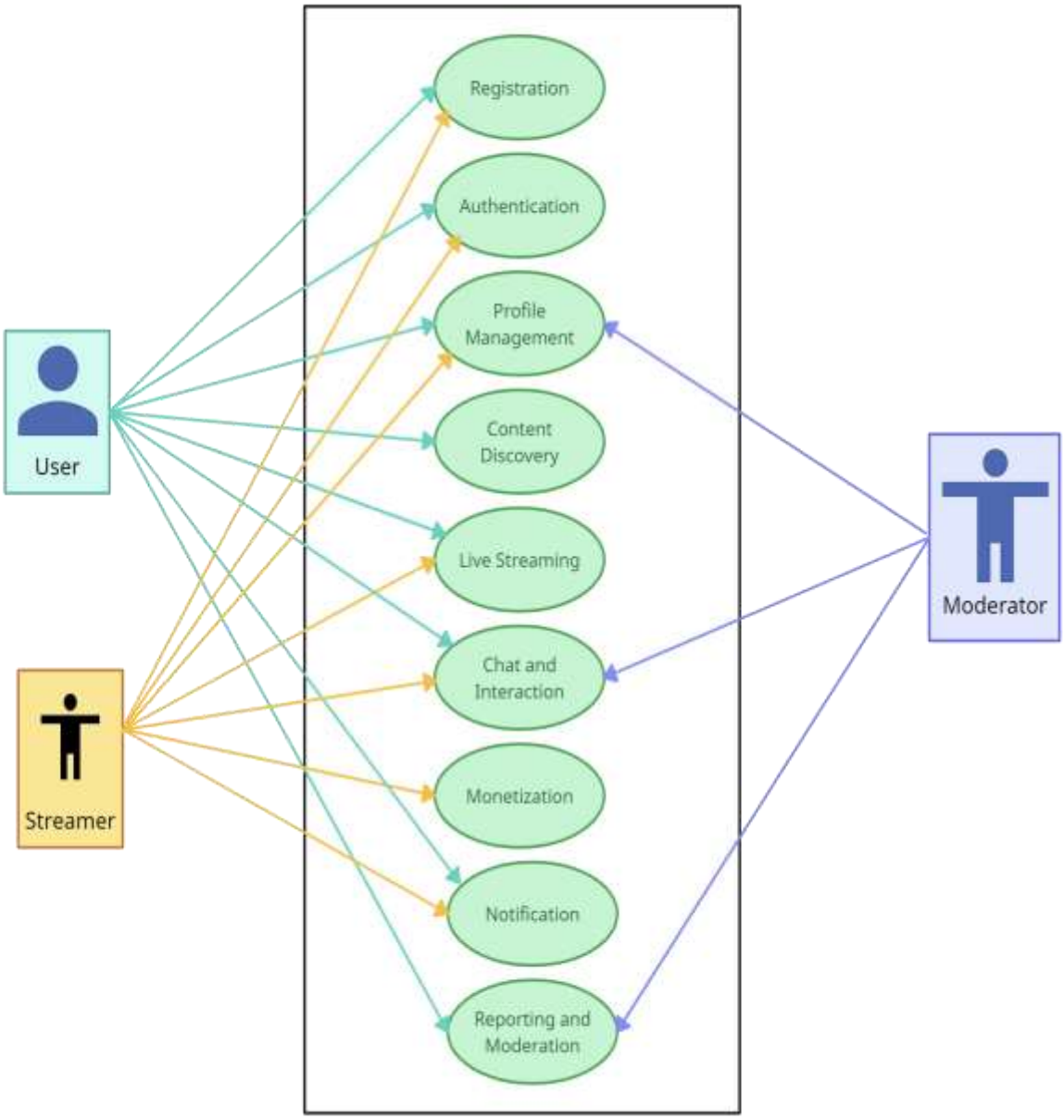
9. Transparency and Communication: Agile methodology ensures great transparency and communication with customers. It promotes transparency through regular ceremonies like stand-up meetings, sprint reviews, and retrospectives.

10. Cross-Functional Teams: Agile teams are typically cross-functional, including members with skills in development, testing, design, and more.

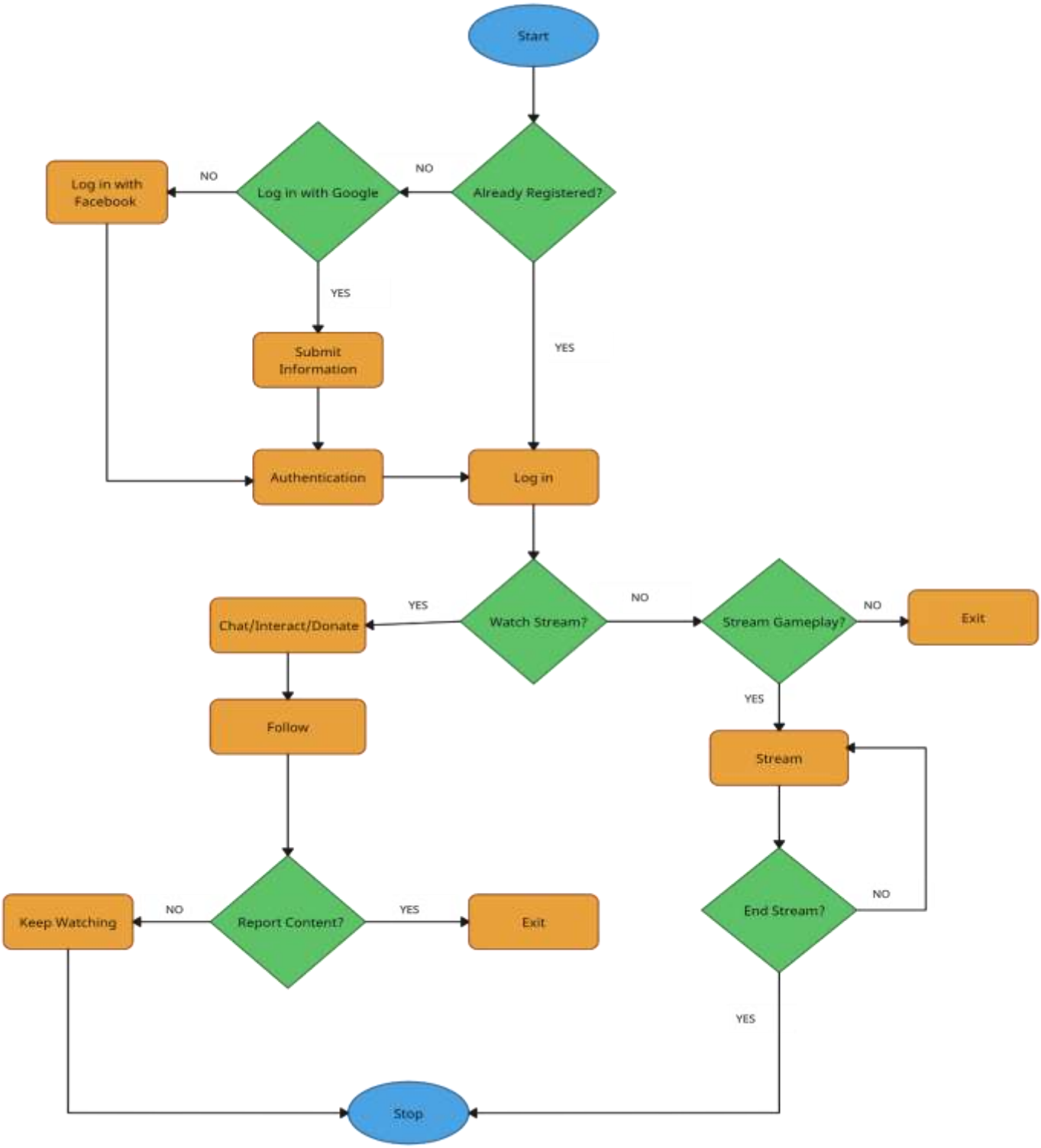
5. System Models/Diagram

Use Case Diagrams, UML diagram and Entity-Relationship Diagrams are displayed in this section.

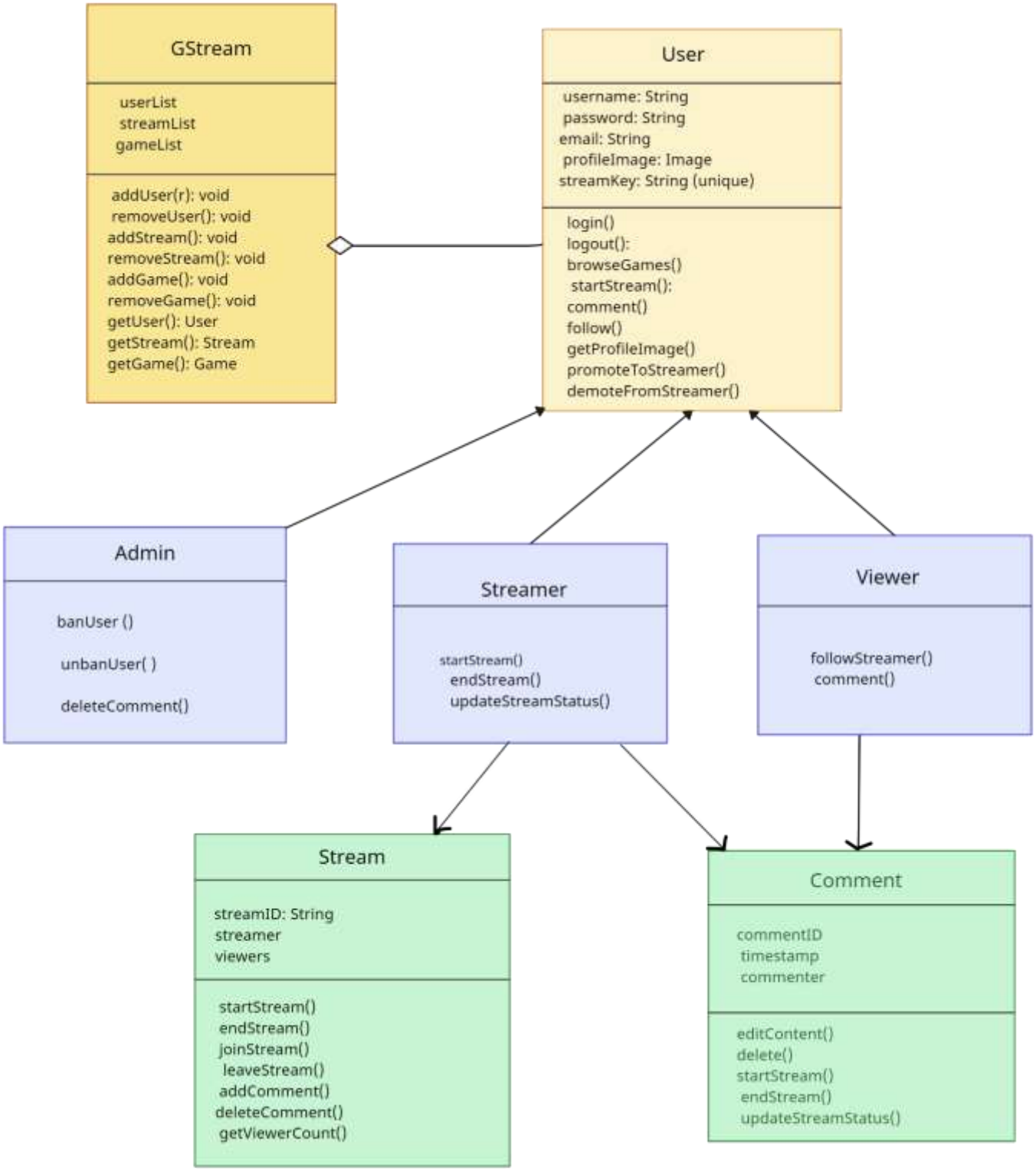
USE CASE DIAGRAM



Flow Chart


Start
Already Registered?
NO
Log in with Google
NO
Log in with Facebook
YES
Submit Information
YES
Authentication
Log in
Watch Stream?
YES
Chat/Interact/Donate
NO
Stream Gameplay?
NO
Exit
YES
Follow
Stream
End Stream?
NO
YES
Report Content?
NO
Keep Watching
YES
Exit
Stop

UML DIAGRAM


GStream
userList
streamList
gameList
addUser(r): void
removeUser(): void
addStream(): void
removeStream(): void
addGame(): void
removeGame(): void
getUser(): User
getStream(): Stream
getGame(): Game
User
username: String
password: String
email: String
profileImage: Image
streamKey: String (unique)
login()
logout():
browseGames()
startStream():
comment()
follow()
getProfileImage()
promoteToStreamer()
demoteFromStreamer()
Admin
banUser ()
unbanUser( )
deleteComment()
Streamer
startStream()
endStream()
updateStreamStatus()
Viewer
followStreamer()
comment()
Stream
streamID: String
streamer
viewers
startStream()
endStream()
joinStream()
leaveStream()
addComment()
deleteComment()
getViewerCount()
Comment
commentID
timestamp
commenter
editContent()
delete()
startStream()
endStream()
updateStreamStatus()

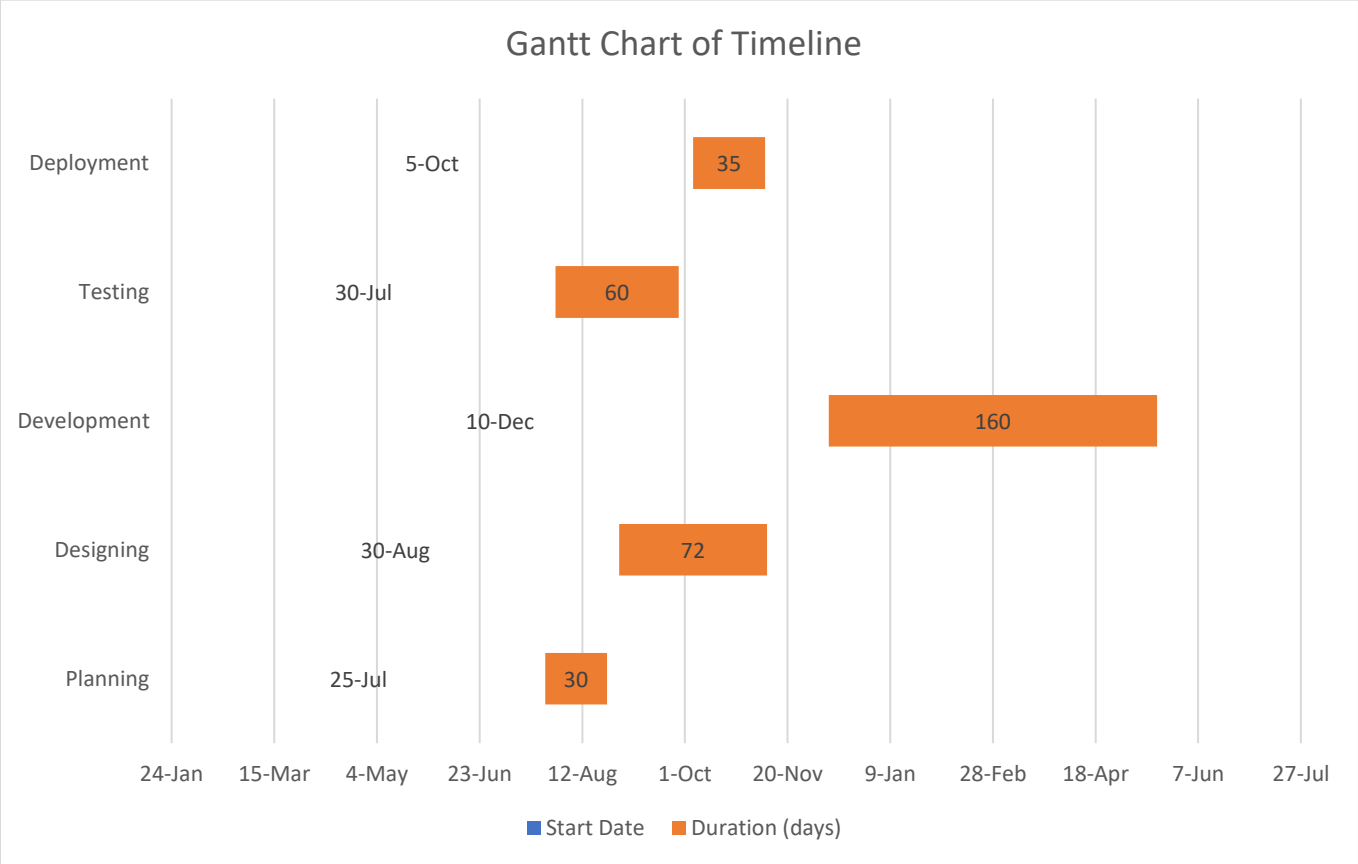
6. Test and Validation

During the project's testing period the tests will be performed are

-Performance Testing.
-Security Testing and Functional Testing
-Usability and Accessibility Testing and
-Platform and Device Specific Testing

7. Project Timeline and Gantt Chart

- The project will be divided into phases: Planning, Design, Development, Testing, and Deployment.



8. Payment Terms

We propose the following payment terms:

Paid on acceptance of this proposal.	10% (10%)
Paid on signing of our Application development agreement.	40% (50%)
Paid at 70% Application Demonstration.	25% (75%)
Paid at completion the Application.	25% (100%)

9. Contact

By Phone:
+8801828867270

Contact Us by Email
HDsector@gmail.com

On our website
www.hdsector.bd

Agreement Signed By:

…… ……

Client Signature

Authority Signature
Hirankur Dewan
Chief Executive Officer
HD SECTOR