**CHAPTER 1**

# INTRODUCTION

Games used to require specialized tools to produce. Now all you need is a browser and a text editor. Even outside of HTML5 games, the time and cost to make a game has dropped so dramatically that people can now build games in hours or days. The indie game developer scene is growing, as are game jams, the online and in-person get-togethers for rapid game making

Each game can be created in a few hours. Will these be your favourite games of the given genre? It’s highly unlikely. These Games demonstrates how to break down genres of games into their basic 2 INTRODUCTION elements. This lays the foundation, puts up the frame, and installs the drywall. In some cases, the author has decorated sparsely. There might be a big hole in the roof and the favourite pictures are hanging on the wall. Don’t hesitate to build a courtyard, install shag carpeting, or plant some ginkgo trees if you want. Take down my pictures. You’ll see where to get all of the materials you need, but it’s your house. Do whatever you want with it. These are your games as soon as you load them up.

## 1.1 Purpose

The main purpose of this project is that this project will provide you seamless gaming environment.

Even on the vary low end system. If the user computer is capable of running an Internet browser then congratulation user is eligible for enjoying the GameX Platform experience

**1.2 About GameX**

Our project (**GameX**) is basically a “Gaming portal”. Here you can play game according to your choice. For playing the game you have to sign up on our page. There is a lot of fun on our page.

So, sit tight and be ready for the awesome adventure.

To launch our platform GameX you just need an Internet browser recommended is Google Chrome or Microsoft edge.

**CHAPTER 2**

# HARDWARE AND SOFTWARE REQUIREMENTS

The GameX is an web gaming platform ,developed on web. So, there are some requirements for running this projectand they are mentioned below-

## 2.1 Software Requirement

OPERATING SYSTEM : WIN 7/10, LINUX. MacOS

DATA BASE : FIREBASE

SOFTWARE : ECLLIPSE , ATOM OR NOTEPAD

FRONT END TOOL : HTML5, CSS

LANGUAGE : JAVASCRIPT, HTML5 , CSS

SCRIPTING LANGUAGE : JAVA SCRIPT, HTML5 , CSS

.

## 2.2 Hardware Requirements

PROCESSOR : Pentinum IV

PROCESSOR SPEED : 2.4GHZ

MONITOR : COLOR MONITOR

RAM : 1 GB

MOUSE : SCROLLING MOUSE

KEY BOARD : MM KEY BOARD

# CHAPTER 3

# MODULES

* **Firebase** (GOOGLE DATABASE)
* **Java Script**(DO THINGS AND HAVE BEHAVIOUR IN A WEBSITE)
* **Canvas 2D**( DEPLOY GRAPHICS ON A WEB PAGE)
* **HTML5** (STRUCTURE OF A WEBSITE)
* **CSS** (STYLING OF A WEBSITE)

# CHAPTER 4

# FIREBASE

## 4.1 About Firebase

Firebase evolved from Envolve, a prior start-up founded by James Templin and Andrew Lee in 2011. Envolve provided developers an API that enables the integration of online chat functionality into their websites. After releasing the chat service, Templin and Lee found that it was being used to pass application data that were not chat messages. Developers were using Envolve to sync application data such as game state in real time across their users. Templin and Lee decided to separate the chat system and the real-time architecture that powered it. They founded Firebase as a separate company in September 2011 and it launched to the public in April 2012. Firebase's first product was the Firebase Realtime Database, an API that synchronizes application data across iOS, Android, and Web devices, and stores it on Firebase's cloud. The product assists software developers in building real-time, collaborative applications.

## 4.2 Feature of Firebase

The features of Firebase are as Follows-

## ****It’s a Realtime Database****

Real-time data is the way of the future. Nothing compares to it. Most databases require you to make HTTP calls to get and sync your data. Most databases give you data only when you ask for it. When you connect your app to Firebase, you’re not connecting through normal HTTP. You’re connecting through a WebSocket. WebSockets are [**much, much faster than HTTP**](http://www.websocket.org/quantum.html). You don’t have to make individual WebSocket calls, because one socket connection is plenty. All of your data syncs automagically through that single WebSocket as fast as your client’s network can carry it.

Firebase sends you new data as soon as it’s updated. When your client saves a change to the data, all connected clients receive the updated data almost instantly.

**It’s Hosting**

Firebase includes an easy-to-use hosting service for all of your static files. It serves them from a global CDN with HTTP/2. And to make your development particularly painless, Firebase hosting utilizes [Superstatic](https://github.com/firebase/superstatic), which you can run locally for all of your testing.

# 4.3 Firebase Pros & Cons

It’s not all roses.

I mean, it’s mostly roses, but watch the thorns.

**Pros**

* Email & password, Google, Facebook, and Github authentication
* Realtime data
* Ready-made api
* Built in security at the data node level
* File storage backed by Google Cloud Storage
* Static file hosting
* Treat data as streams to build highly scalable applications
* Don’t worry about your infrastructure!

## Cons

* Limited query abilities due to Firebase’s data stream model
* Traditional relational data models are not applicable to NoSQL; therefore, your SQL chops will not transfer
* No on-premise installation

**CHAPTER 5**

**JavaScript**

JavaScript is a programming language that adds interactivity to your website. This happens in games, in the behavior of responses when buttons are pressed or with data entry on forms; with dynamic styling; with animation, etc. This article helps you get started with JavaScript and furthers your understanding of what is possible

## 5.1 About JavaScript

[**JavaScript**](https://developer.mozilla.org/en-US/docs/Glossary/JavaScript) ("JS" for short) is a full-fledged [**dynamic programming language**](https://developer.mozilla.org/en-US/docs/Glossary/Dynamic_programming_language) that can add interactivity to a website. It was invented by Brendan Eich (co-founder of the Mozilla project, the Mozilla Foundation, and the Mozilla Corporation).

JavaScript is versatile and beginner-friendly. With more experience, you'll be able to create games, animated 2D and 3D graphics, comprehensive database-driven apps, and much more!

JavaScript itself is relatively compact, yet very flexible. Developers have written a variety of tools on top of the core JavaScript language, unlocking a vast amount of functionality with minimum effort. These include:

* Browser Application Programming Interfaces **(**[**APIs**](https://developer.mozilla.org/en-US/docs/Glossary/API)**)** built into web browsers, providing functionality such as dynamically creating HTML and setting CSS styles; collecting and manipulating a video stream from a user's webcam, or generating 3D graphics and audio samples.
* Third-party APIs that allow developers to incorporate functionality in sites from other content providers, such as Twitter or Facebook.
* Third-party frameworks and libraries that you can apply to HTML to accelerate the work of building sites and applications.

**5.2** **Question related to game development in JavaScript.**

**5.2.1** **can JavaScript be used to makes games?**

JavaScript can be used to make games using a variety of platforms and tools. Both 2d and 3d libraries can be used in combination with JavaScript to create fully-fledged games in the browser or external game engine platforms.

## 5.1.2 Is JavaScript good for creating games?

The answer to this heavily relies on your own criteria of what constitutes a good game creation language. It should be noted that the majority of JavaScript game development focuses on creating games used for the browser. This in itself creates some limitations. Common gaming aspects such as **collision detection** or other rendering tasks typically require heavy calculations using GPU computation. However, since we’re confined to the browser the computation power is not as powerful as other game rendering engines and technologies.

As a result, JavaScript is currently not a sufficient tool for making AAA games. The lack of rendering pipelines, true OOP and advanced memory management hinders it for these types of titles.

## 5.1.3 Is it easy to make game with JavaScript?

**Javascript** is one of the most popular languages for creating games. It naturally works well with HTML and CSS and is well suited for online games. Javascript can be used on both the back end and front end of web development and has a huge, helpful online community as well as an enormous number of frameworks. Javascript has proven to be an extremely versatile language, and with the help of libraries can be used for iOS and Android Apps, desktop apps, and hardware. With many free online tutorials and a massive GitHub presence, Javascript is one of the easiest languages to learn. It can also be used for scripting with Unity3D.

Common gaming aspects such as collision detection or other rendering tasks typically require heavy calculations using GPU computation. However, since we’re confined to the browser the computation power is not as powerful as other game rendering engines and technologies. Additionally, since the majority of JavaScript game frameworks are also targetting the browser. These games will essentially not be native. However, just because the majority of the games are targetting the browser, doesn’t delegitimize the scope and capabilities that some JavaScript games are striving to achieve.

**CHAPTER 6**

**Canvas 2D**

The HTML <canvas> element is displayed as a rectangular object on a web page:

Fig. 6.1 Rectangular Object

## 6.1 About HTML Canvas

The <canvas> element is perfect for making games in HTML.

The <canvas> element offers all the functionality you need for making games.

Use JavaScript to draw, write, insert images, and more, onto the <canvas>.

## 6.1.1 .getContext("2d")

The <canvas> element has a built-in object, called the getContext("2d") object, with methods and properties for drawing.

## 6.2 Browser Support

The numbers in the table specify the first browser version that fully supports the <canvas> element.

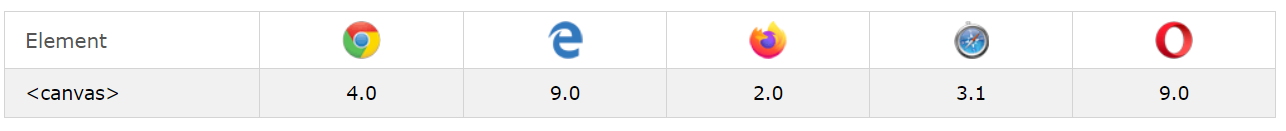


Fig. 6.2 Browser Support

## 6.3 Get Started

A canvas is a rectangular area on an HTML page. By default, a canvas has no border and no content.

The markup looks like this:

**<canvas id="myCanvas" width="200" height="100"></canvas>**

**Note:** Always specify an id attribute (to be referred to in a script), and a width and height attribute to define the size of the canvas. To add a border, use the style attribute.

Here is an example of a basic, empty canvas:

Fig. 6.3 Empty Canvas

## 6.4 Add a JavaScript

After creating the rectangular canvas area, you must add a JavaScript to do the drawing.

Here are some examples:

### **6.4.1 Draw a Line**

Fig. 6.4 Draw A line in a canwas

### **Code for Drawing line.**

<script>  
var c=document.getElementById("myCanvas");  
var ctx=c.getContext("2d");  
ctx.moveTo(0, 0);  
ctx.lineTo(200, 100);  
ctx.stroke();  
</script>

### **6.4.2 Draw a Text**

HelloWorld

Fig .6.5 Writing a Text in a Canvas

### **Code for Drawing a Text.**

<script>  
var c=document.getElementById("myCanvas");  
var ctx=c.getContext("2d");  
ctx.font = "30pxArial";  
ctx.fillText("HelloWorld", 10, 50);  
</script>

**CHAPTER 7**

**HTML 5**

**7.1 About HTML**

HTML is the standard markup language for creating Web pages.

## 7.1.1 What is HTML?

* HTML stands for Hyper Text Markup Language
* HTML is the standard markup language for creating Web pages
* HTML describes the structure of a Web page
* HTML consists of a series of elements
* HTML elements tell the browser how to display the content
* HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.

## 7.1.2 A Simple HTML Document

<!DOCTYPE html>  
<html>  
<head>  
<title>PageTitle</title>  
</head>  
<body>  
  
<h1>MyFirstHeading</h1>  
<p>Myfirstparagraph.</p>  
  
</body>  
</html>

# 7.2 HTML Editor

A simple text editor is all you need to learn HTML.

## 7.2.1 Learn HTML Using Notepad or TextEdit

Web pages can be created and modified by using professional HTML editors.

However, for learning HTML we recommend a simple text editor like Notepad (PC) or TextEdit (Mac).

We believe in that using a simple text editor is a good way to learn HTML.

Follow the steps below to create your first web page with Notepad or TextEdit.

## Step 1: Open Notepad (PC)

**Windows 8 or later:**

Open the **Start Screen** (the window symbol at the bottom left on your screen). Type **Notepad**.

**Windows 7 or earlier:**

Open **Start** >**Programs >** **Accessories >** **Notepad**

## Step 1: Open TextEdit (Mac)

Open **Finder > Applications > TextEdit**

Also change some preferences to get the application to save files correctly. In **Preferences > Format >**choose**"Plain Text"**

Then under "Open and Save", check the box that says "Display HTML files as HTML code instead of formatted text".

**Then open a new document to place the code.**

**CHAPTER 8**

**CSS**

# 8.1 CSS Introduction

It provide Styling to the HTML.

## 8.1.1 What is CSS?

* **CSS** stands for **C**ascading **S**tyle **S**heets
* CSS describes **how HTML elements are to be displayed on screen, paper, or in other media**
* CSS **saves a lot of work**. It can control the layout of multiple web pages all at once
* External stylesheets are stored in **CSS files**

**8.1.2 Why Use CSS?**

CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

**CSS Example**

body {  
  background-color: **lightblue;**  
}  
h1 {  
  color: **white;**  
  text-align: **center;**  
}  
p {  
 font-family: **verdana;**  
  font-size: **20px;**  
}

## 8.2 CSS Syntax

A CSS rule-set consists of a selector and a declaration block:



Fig 8.1 CSS Syntax

The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a CSS property name and a value, separated by a colon.

Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.

Example

In this example all <p> elements will be center-aligned, with a red text color:

p {  
  color: **red**;  
  text-align: **center**;  
}

**CHAPTER 9**

**SCREENSHOTS**

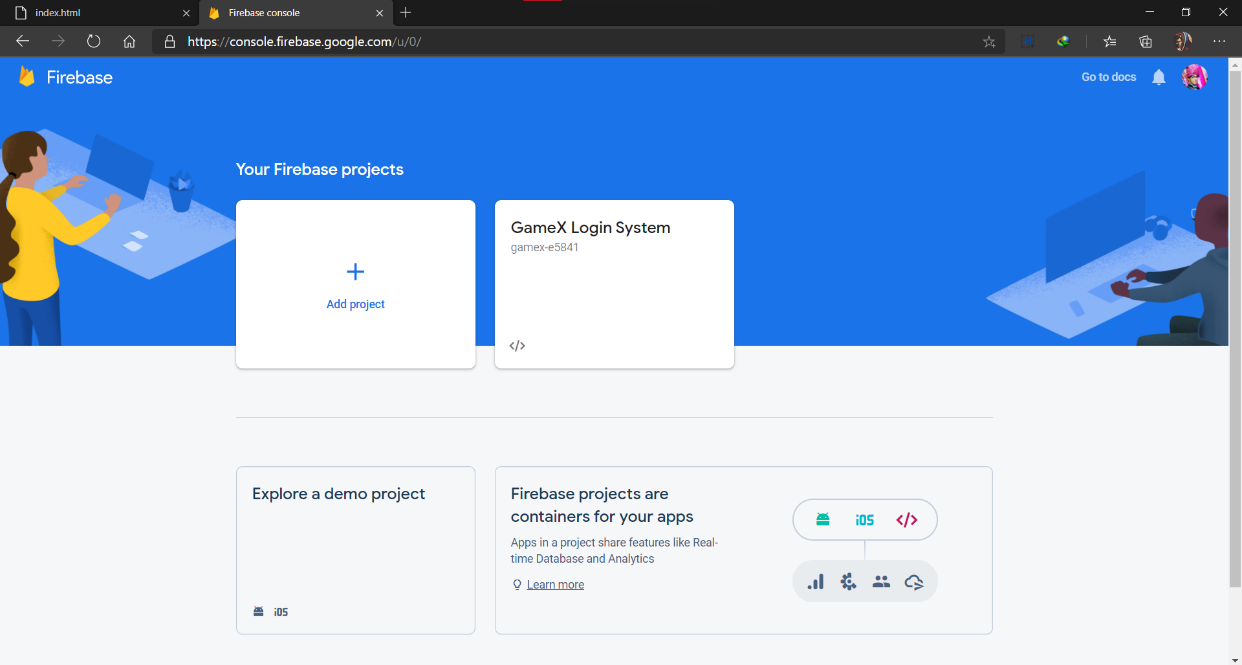


Fig 9.1 GameX Firebase

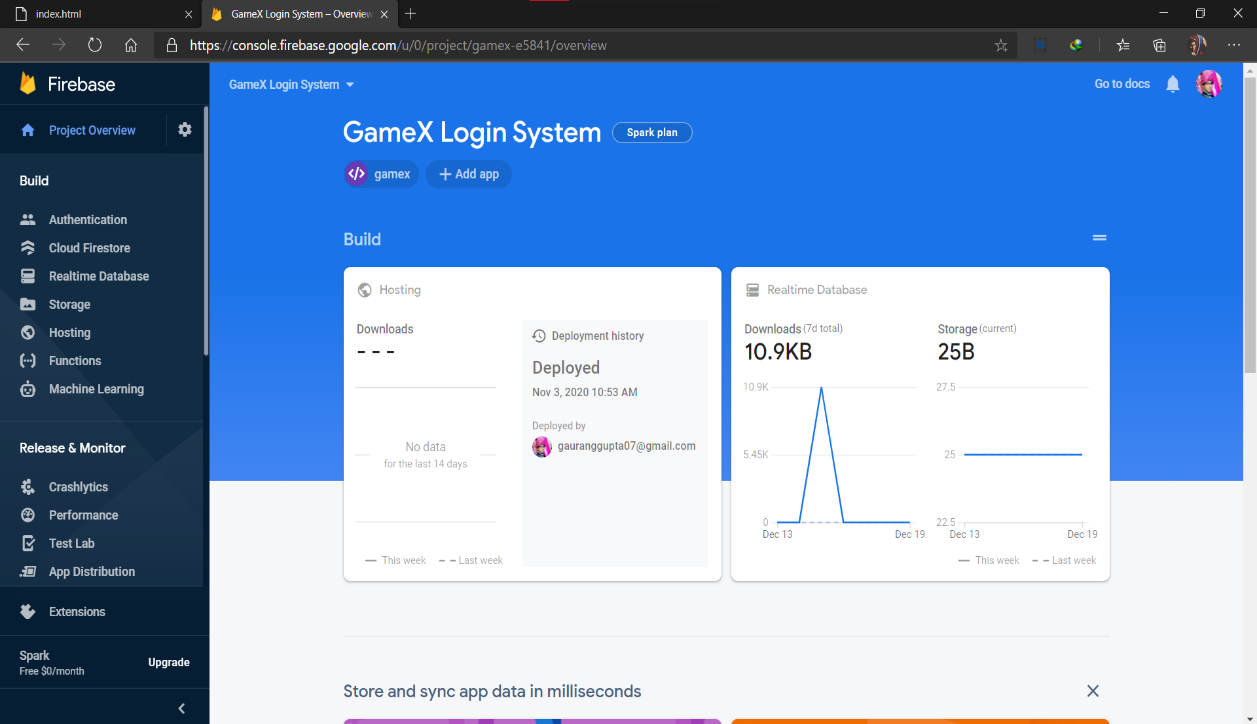


Fig 9.2 GameX Firebase Login System

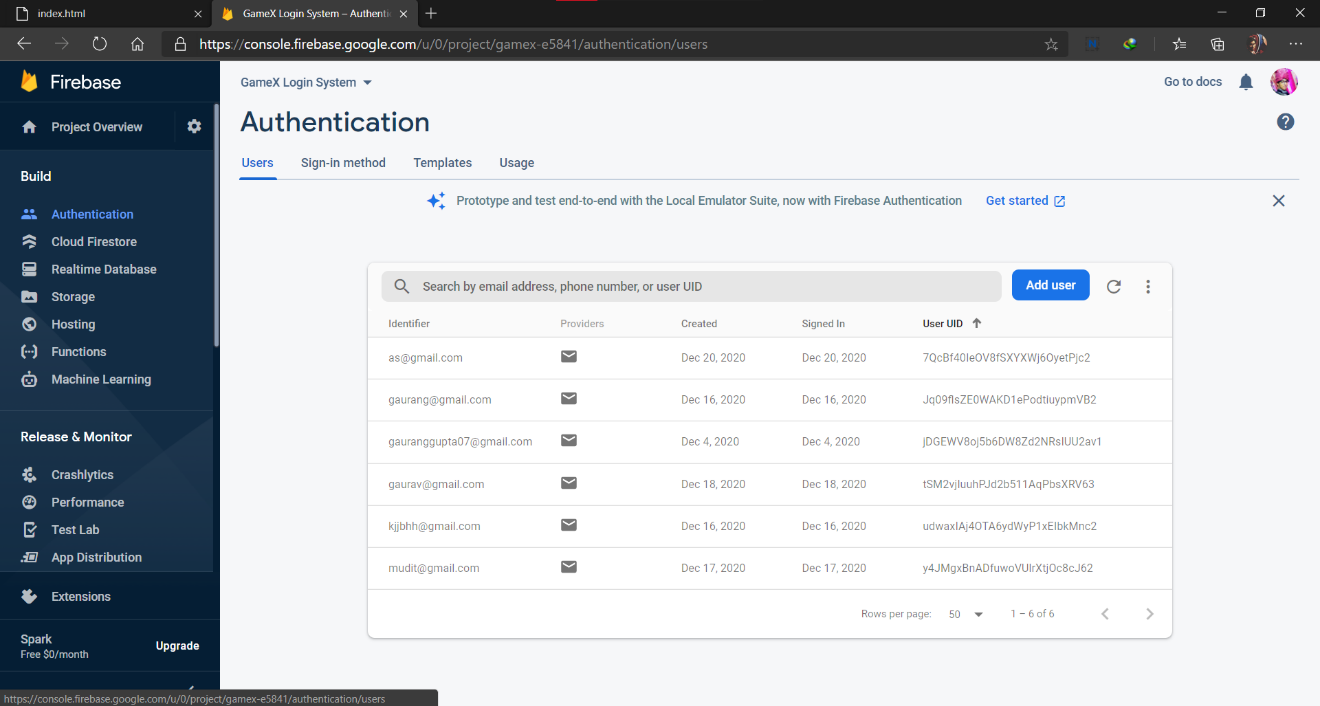


Fig 9.3 GameX Authentication Page

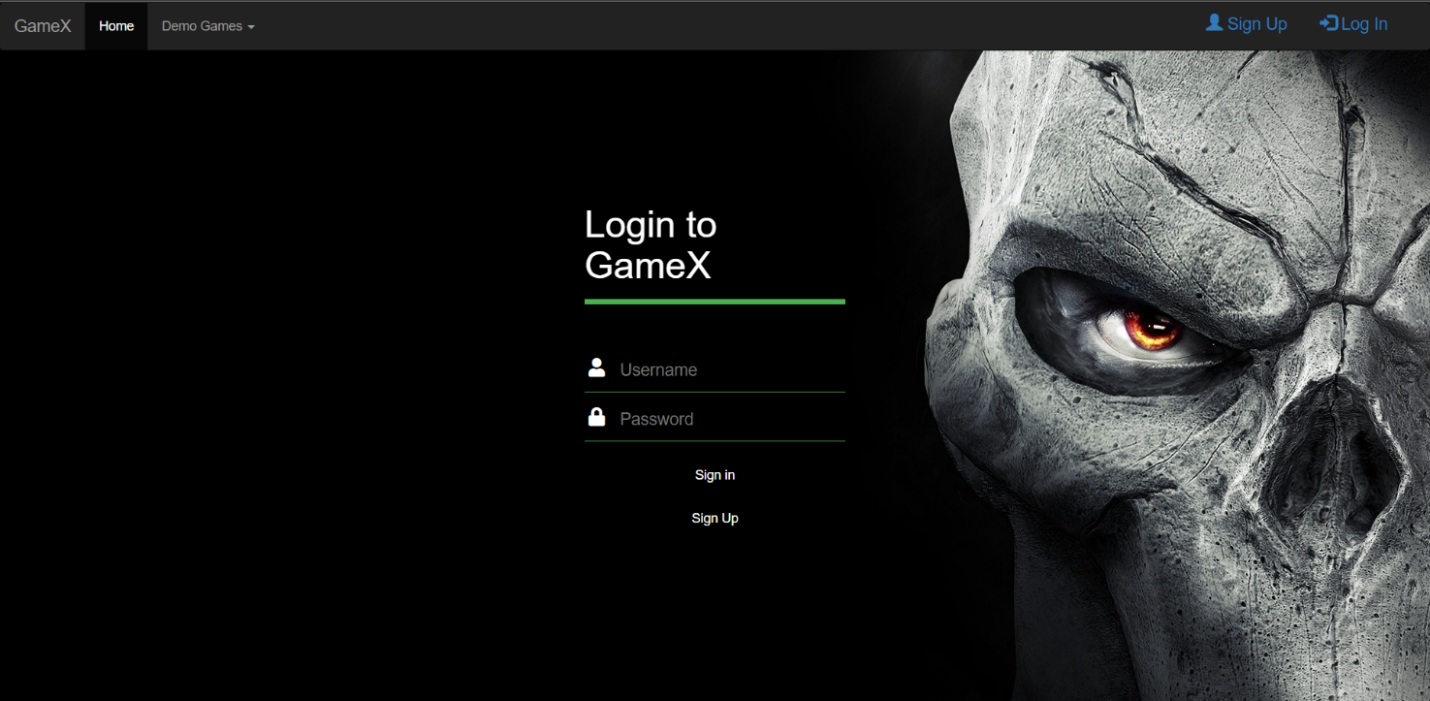


Fig 9.4 GameX Login Page

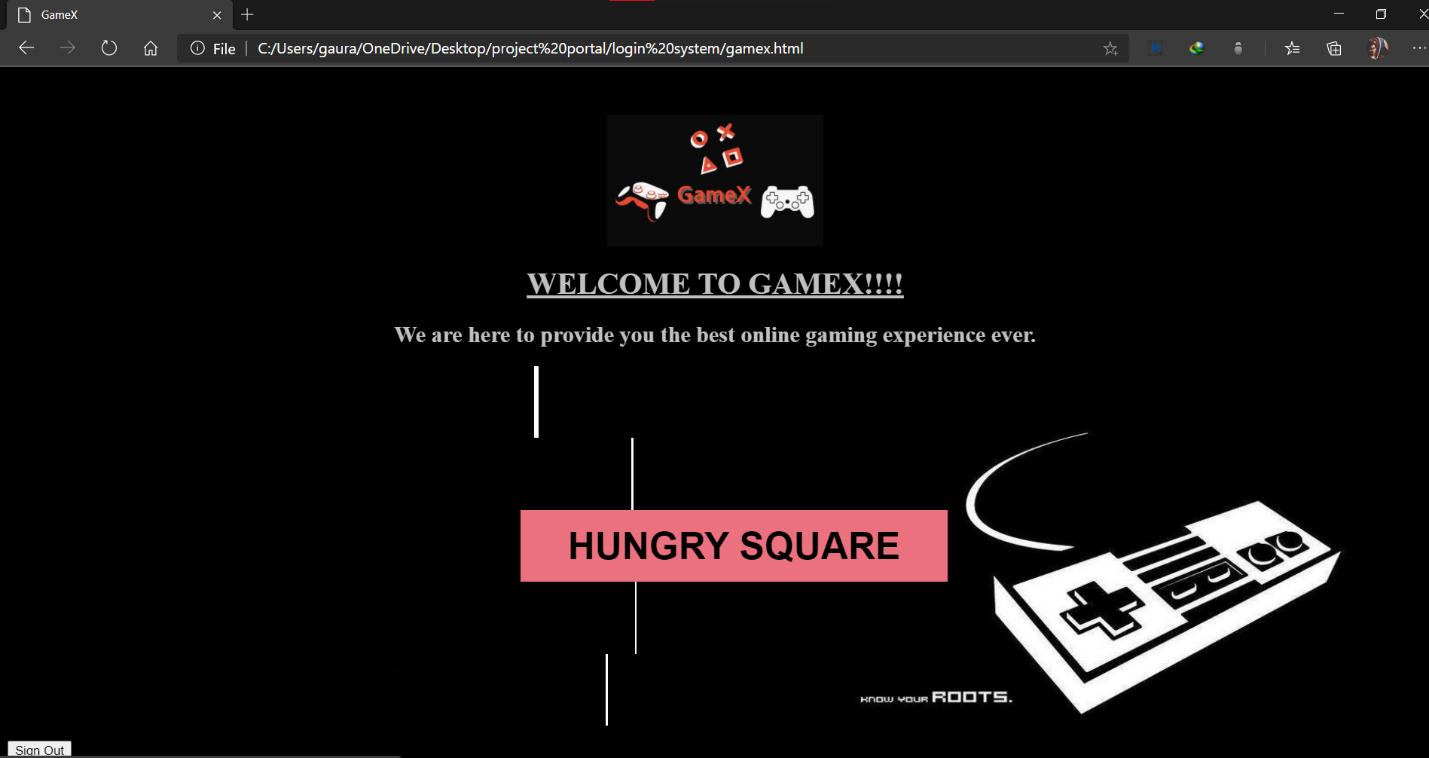


Fig 9.5 GameX Welcome Page



Fig 9.6 Square Shoot Game

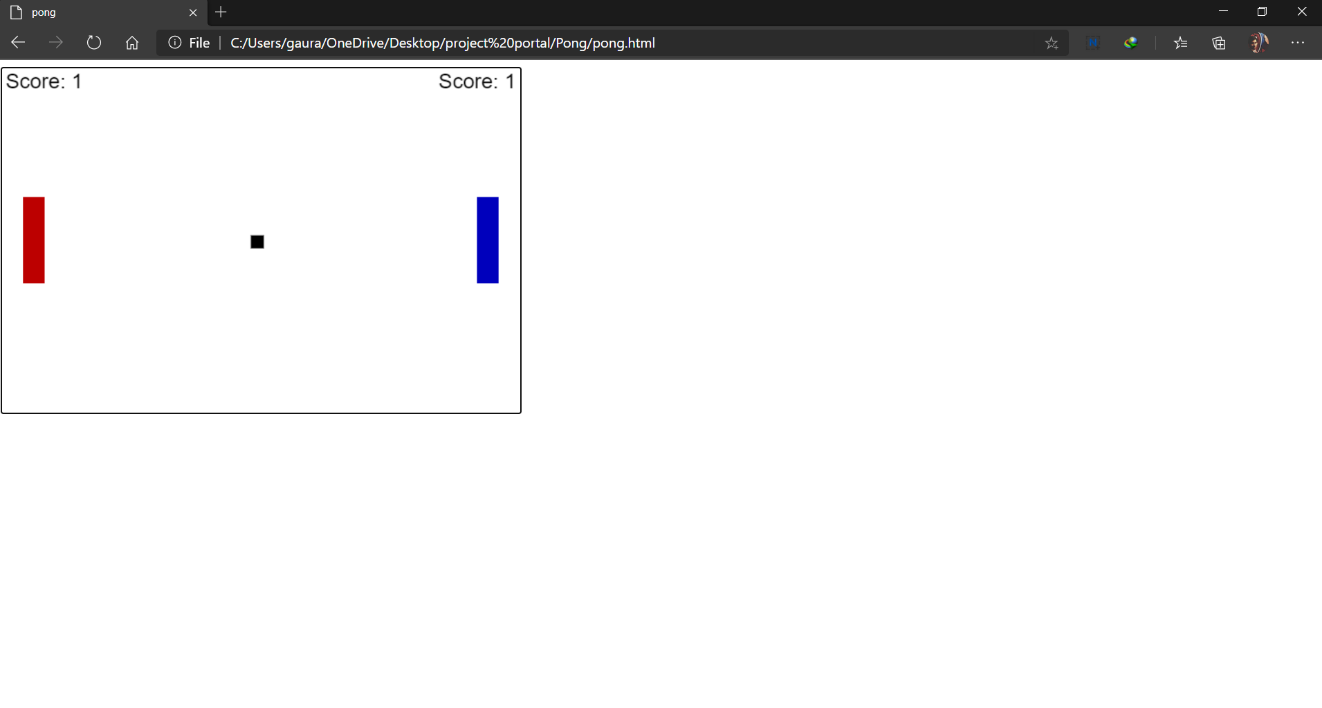


Fig 9.7 Pong Game

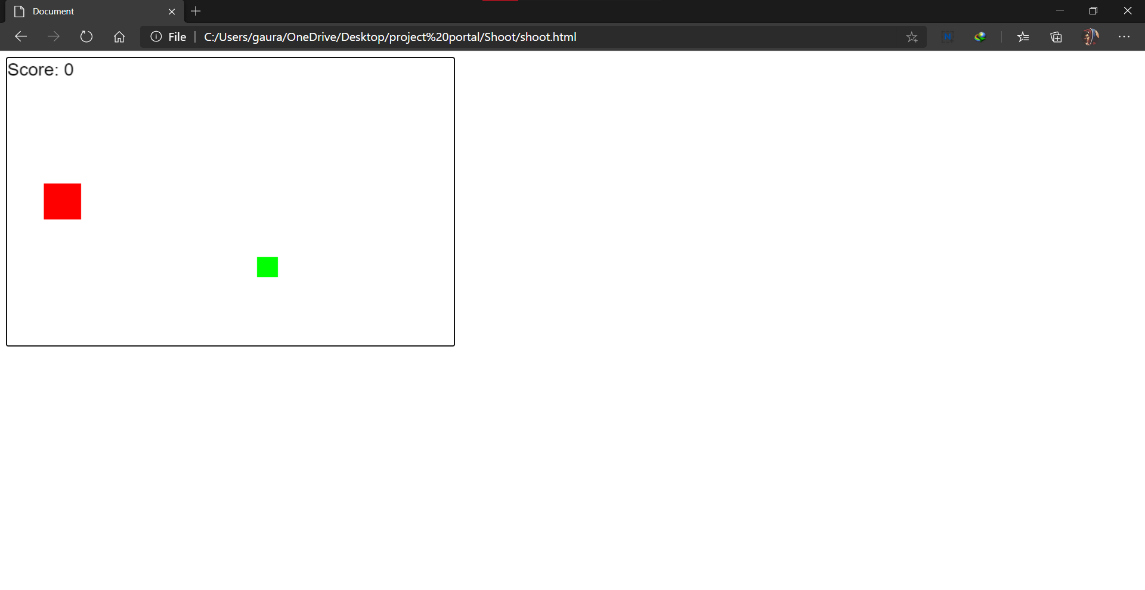


Fig 9.8 Hungry Square Game

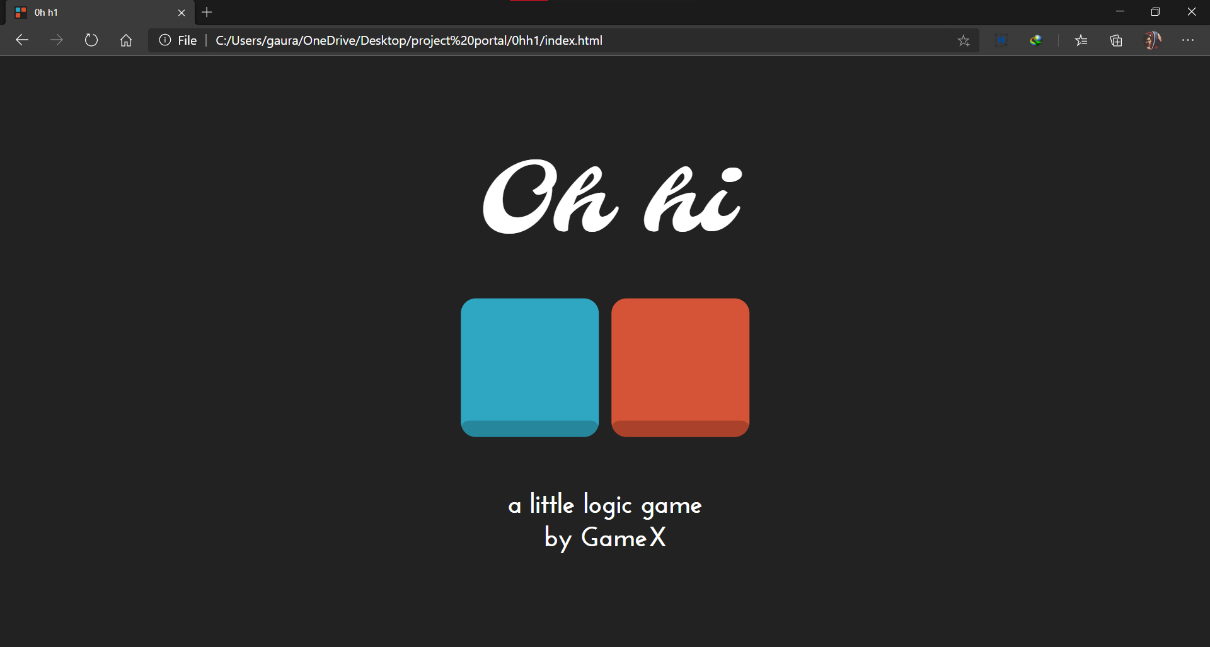


Fig 9.9 OHH1

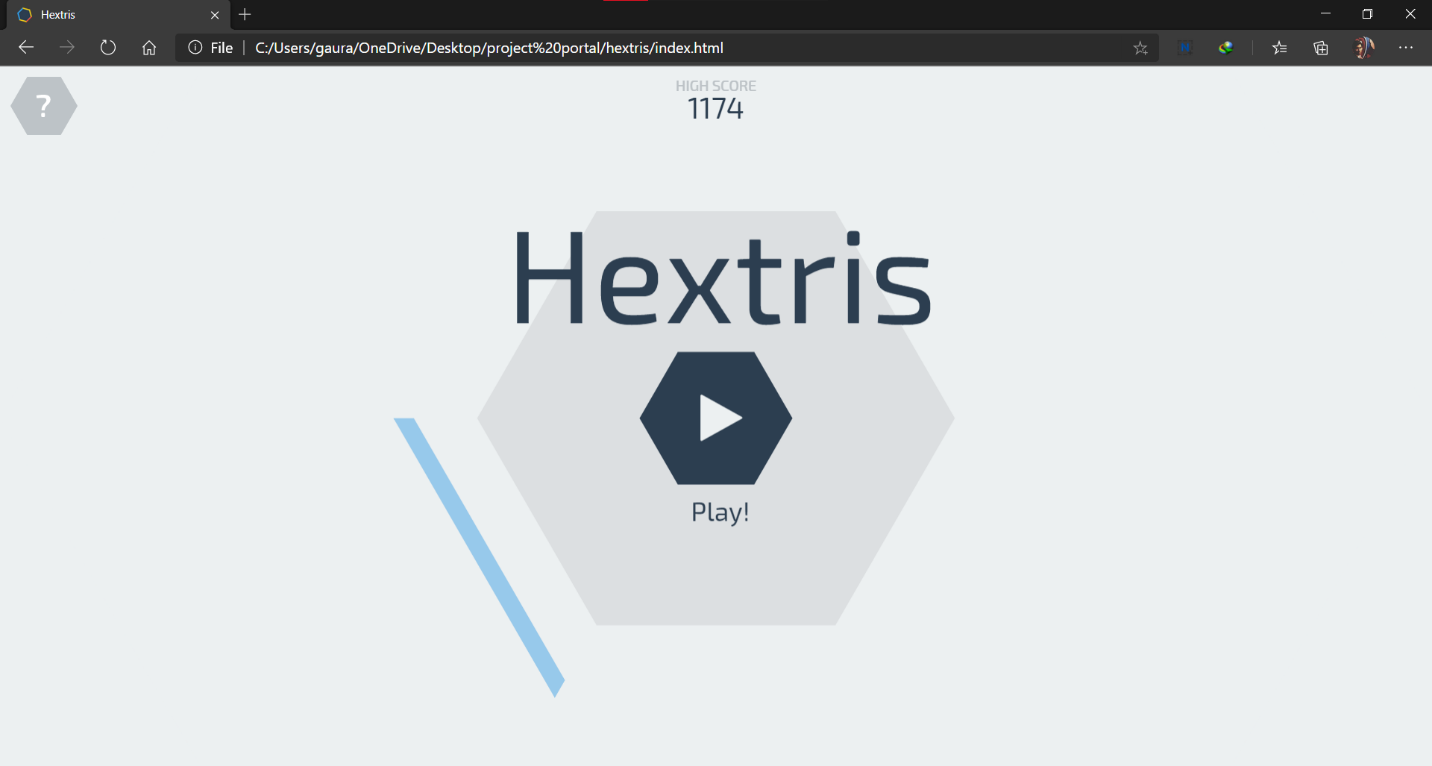


Fig 9.10 Hextris

# CHAPTER-10

# CONCLUSION

## 10.1 Conclusion

Over the past 11 chapters, you have been immersed in the world of HTML5 Canvas. We have given you dozens of examples and applications to work from and through so that you can start building your own creations. From simple text displays to high-performance games, we have shown you many ways to bring some of the magic of previous RIA (Rich Internet Application) technologies into the plug-in-less browser experience.

We offered many strategies for integrating Canvas with other HTML5 technologies, as well as techniques for handling text, displaying graphics, scrolling bitmaps, creating animation, detecting multiple types of collisions, embedding and manipulating video, playing music, handling sound effects, creating user interfaces, optimizing code, and preparing apps for the mobile web and Windows 8. We even introduced you to the future of 3D and multiuser applications directly in the web browser and showed you how to get started creating an object model for the HTML5 Canvas. However, the true future is up to you. HTML5 and Canvas are dynamic topics that are still in a rapid state of change and adoption. While this book is a good starting point, you will need to keep abreast of new changes to the technology. Visit our website for news and updates on HTML5 Canvas.

There is a real paradigm shift occurring right now on the Web. For most of the first decade of the 21st century, Java, Flash, Silverlight, and other plug-in RIA technologies dominated application development and design. At the time, there appeared to be no better solution for the development of rich applications in a web browser than to bolt on technology that was not native to the browser.

The emergence of the “connected apps” culture is changing this. Every platform—from tablets and phones to TVs, e-readers to tablets, wireless printers to desktop PCs—is targeted for web-enabled applications sold or distributed through an app store. In many ways, these apps are replacing RIA applications or, at the very least, offering a compelling new platform for their development and distribution.

**CHAPTER 11**

# FUTURE SCOPES

## 11.1 Future Scopes

Seeing the New Era, we can easily understand there is infinitely many future Scopes in the Gaming industry. Some of the Features that We Will implement in future are as follows

* Develop a real time leaderboard for online multiuser gaming

* OTP verification on user login
* More Advanced Game
* Special Username for each game .
* Mail verification on user login
* Direct Google or facebook SignIn

**CHAPTER 12**

**BIBLIOGRAPHY**

1. http://en.wikipedia.org
2. http://www.thenewboston.com
3. http://developer.android.com
4. http://www.stackoverflow.com
5. http://www.coderzheaven.com