

Brock University

Computer Science Department  
COSC 4P02 – Software Engineering 2

Final Report

**HabitForge**

[www.habitforge.ca](http://www.habitforge.ca)



Authors:

Abhi Patel – 6897334 – [ap19wf@brocku.ca](mailto:ap19wf@brocku.ca)  
Ameen Khawaja – 6935688 – [ak19nu@brocku.ca](mailto:ak19nu@brocku.ca)  
Zakir Raza – 6834923 – [zr19zt@brocku.ca](mailto:zr19zt@brocku.ca)  
Nico McFarlane – 7001811 – [nm20lw@brocku.ca](mailto:nm20lw@brocku.ca)  
Rafael Bocsa – 7035801 – [rb20qf@brocku.ca](mailto:rb20qf@brocku.ca)  
Ahmed Yaser – 7063944 – [am20gy@brocku.ca](mailto:am20gy@brocku.ca)

Professor: Dr. Naser Ezzati-Jivan

## Introduction

The HabitForge team is so excited to showcase the final product through the immense progress we have made as a team throughout the last four months. We will discuss each group member's contributions to HabitForge, along with an updated requirements document, installation manual, and test documents/results. We would also like to announce that you can view our software at [www.habitforge.ca](http://www.habitforge.ca).

## Abhi Patel [Scrum Master + Developer]

I was the scrum master and developer for HabitForge. As a scrum master I planned out the sprints for the four months of development. The planned sprints can be found on the provided Trello board. I also set up the code base and GitHub repo for the team. Since we were following the Agile methodology, we had weekly meetings and bi-weekly sprint overview meetings that were all facilitated by me. I also worked on all the UI designs that were used by the development team. As a developer I worked on many features including:

- Initial Codebase: Setting up Next.JS, Tailwind CSS, Firebase project, providing environment variables. Created the routes for the app as well and protecting some routes. Like dashboard is only accessed if a user is logged in, if they are not and try to access the dashboard by typing it in the URL, they are taken to the log in page first. If a user is logged in and goes to the log in page again then they are routed back to the dashboard.
- Dashboard
  - Creating Habits, Storing Habits, Tracking Habits, Editing habits, Deleting Habits,
  - Group Habits, inviting members by email, accepting invites, showing the members on group habits.
  - Connecting analytics components to the data from Firebase.
  - Overall, the process of the dashboard was done in iterations planned out by the sprints. First, I created the front-end for creating habits and how habits will be showcased to the user following the designs made in Figma. Then I implemented the firebase database so that the habit created is stored properly and can be tracked. I created cloud functions that reset the habits based on their goals. Once the habit creation was tested properly and looked stable, then I moved on to creating Group Habits. I followed the same process for that as well. Along the way I also created a notifications collection for users that have invites/ notifications. This would be used to send group habit invites to users.
- Leaderboards
  - Created the front-end of the leaderboards page.
  - Connected the frontend to the firebase leaderboards collection.

- For the leaderboards the steps were simple, was able to create the frontend by following the Figma designs and after that I created a cloud function that pulled in all users and their rank and stores them in the leaderboards collection. This was then used to populate the frontend.
- Firebase
  - Structured the database including, user collection, habits collection, each individual habit, groups collection. The structure was created by following the user journey through out the app in the Figma. I analyzed at every step what data would be required to be stored and pulled in and that is how I was able to structure the database. I took rough notes on notion for how that would look and here's a screenshot for that.

```

uid: String!
  displayName: String
  email: String
  uid: String
  createdAt: String
  level: Int //not sure if keeping
  totalHabits: Int
  habitCoins: Int
  rank: Int

Habits: {
  HUID: String {
    name: String
    goal: Int // if times-> 1 - 5 if minutes -> 3 hours max
    repeat: String //Daily | Weekly | Monthly
    startDate: String //start counting from
    completed: Boolean //this will reset at every goal interval
    skipped: Boolean //this will reset at every goal interval
    failed: Boolean //this will reset at every goal interval
    totalCompleted: Int
    totalSkipped: Int
    totalFailed: Int
    streak: Int //when a habit is completed do the following -> check if current date is consecutive to the lastCom
    //if it is then increase streak by 1, if failed/skipped/not consecutive -> reset streak to 0
    lastCompletedDate: String
    groupId: String //foreign key referring to the group table
    // if a habit has a groupId then we showcase it in the group section
  }
}

Groups: {
  groupId: String {
    groupName: String //Take users display name and add group after Ex. Abhi Patel's Group
    HUID: String
    Members: {
      uid: String!
      creator: Boolean "true"
      status: String "pending" | "accepted" | "declined"
    }
  }
}

Notifications: {
  notifID: String {
    timestamp: String
    title: String
    senderUID: String
    HUID: String
    groupId: String
    status: String "pending" as default | "accepted" | "rejected"
  }
}

```

## Ahmed Yaser [Product Owner + Developer]

The role I claimed at HabitForge was being a Product Owner as well as a developer.

### My Duty as a Product Owner:

I made sure all the sprints for the project were meticulously planned out as well as progress being made in a timely manner. The product vision was defined to the team and made sure that with each sprint completed, the vision remained consistent. I also set up team meetings to ensure that all of us communicated and bounced ideas off of each other during these meetings. Whenever we would encounter a roadblock or code issue we made sure to discuss collectively.

I also made sure that each team member had equal work load or in some cases team up together to work on a single feature/function within the project.

### **My Duty as a Developer:**

#### **1. Research**

- a. Sprint 1 was all about researching why we needed the functions to be developed. So, I was in charge of handling the audio design for the project. The research mainly focused on the psychological factor of how humans reacted toward different sounds and how different sounds reflected different moods. This was accomplished by reading several articles and research papers.

#### **2. Implementation**

- a. Implementing the code meant that I had to learn the basics of React and FireBase to better understand how to store these sound samples in a server and how to implement the functionality of these samples throughout the webpage.
- b. I have also implemented a mute button as a feature.

#### **3. Landing Page**

- a. The landing page was designed by Abhi Patel (Scrum Master) and implemented by both Zakir and I. This was where the React and TypeScript learning curve took place between me and Zakir.
- b. We also ensured that design was responsive so that mobile users could get the best version of the project.
- c. Error solving was my main factor that I contributed in for the landing page while Zakir collaboratively wrote the code for the UI.

Overall, during the development phase of HabitForge, I was able to define the product vision to all the team members. Helped them to prioritize their weekly goals by designing the sprints. Made sure to give out feedback during the weekly meetings and ensured that communication was held consistently. By aligning the requirements and vision for the project, I can say that the team collectively brought it to life.

## **Ameen Khawaja [Project Manager + Developer]**

My role at HabitForge was being both a Project Manager and Developer. First, I'll discuss my contributions from a Project Manager perspective, followed secondly by my contributions from a developer perspective. Down below, for every feature/task I did, I've also provided a reference link showcasing it.

To begin, as a Project Manager, I was mainly responsible for the following:

- **Writing progress reports & assisting with the proposal**
- **Setup project documentation instructions**
  - [Link to Instructions of Running HabitForge Locally](#)
- **Wrote down sprint notes from weekly meetings that took place on Discord.**
  - The sprint notes documentation discusses in depth what we have accomplished each week, what we discussed as a group, any technical issues that came up, the tasks each user is assigned, and what our plans for the upcoming sprint was.
    - [Link to Sprint Notes Documentation](#)
- **Facilitated risk management**
  - Before beginning the development of HabitForge, I ensured we had a separate development and main branch, along with a separate branch for every single feature. This is to ensure that if a teammate accidentally pushes broken code, it will not disrupt our live production code.
- **Ensured all features requested by the Product Owner are delivered in the final version of the Software that we presented**
  - This includes making user stories are implemented, project objectives and requirements are complete, and the roadmap ([Trello Board](#)) is being followed correctly.

Next, I'll discuss my accomplishments and contributions to HabitForge from a developer perspective.

- **Speech-to-text implementation for adding a habit**
  - First phase to implementing this was researching the different APIs available. The most compelling options were Google Cloud's Speech-to-text API, AssemblyAI's API, and Mozilla's SpeechRecognition API. Both Google and AssemblyAI's offering had a premium cost attached to it and is a pay-as-you-go system, so I decided not to use that. I settled on using Mozilla's which is free and works as intended for our use case. The downsides to Mozilla's API are that it is unable to differentiate between words and integer values and lacks context, for example, if a user says "One", It wouldn't know to detect it as "one" or "1" depending on the context, which is problematic for our use case, but more on this will be explained later.
  - In the second phase of implementing speech-to-text, I managed to get it working such that it picks up the user's dialogue and displays it on the webpage.
  - In the third phase of implementing speech-to-text, I integrated it to work with the front-end user interface for adding a habit. However, the SpeechRecognition APIs began to show its limitations for the "Goal /per day" section, as that field only takes integer values (see photo below). The API would fill the field in as a string such as "two per/day", which caused some issues with storing the habits properly. GitHub: [Speech-to-text branch 1](#)

Goal /per day

1

1 ✓

2

3

4

5

- In the final phase of implementation for speech-to-text, many of the bugs were fixed out such as the microphone infinitely listening to the user, the form field now converts words to integers, and only detects a habit name of 25 or less characters. Additionally, speech-to-text works on mobile phones too – in my testing, I tried it on an iPhone and a Samsung. The code can be seen here: [Speech-to-text new updated branch](#)

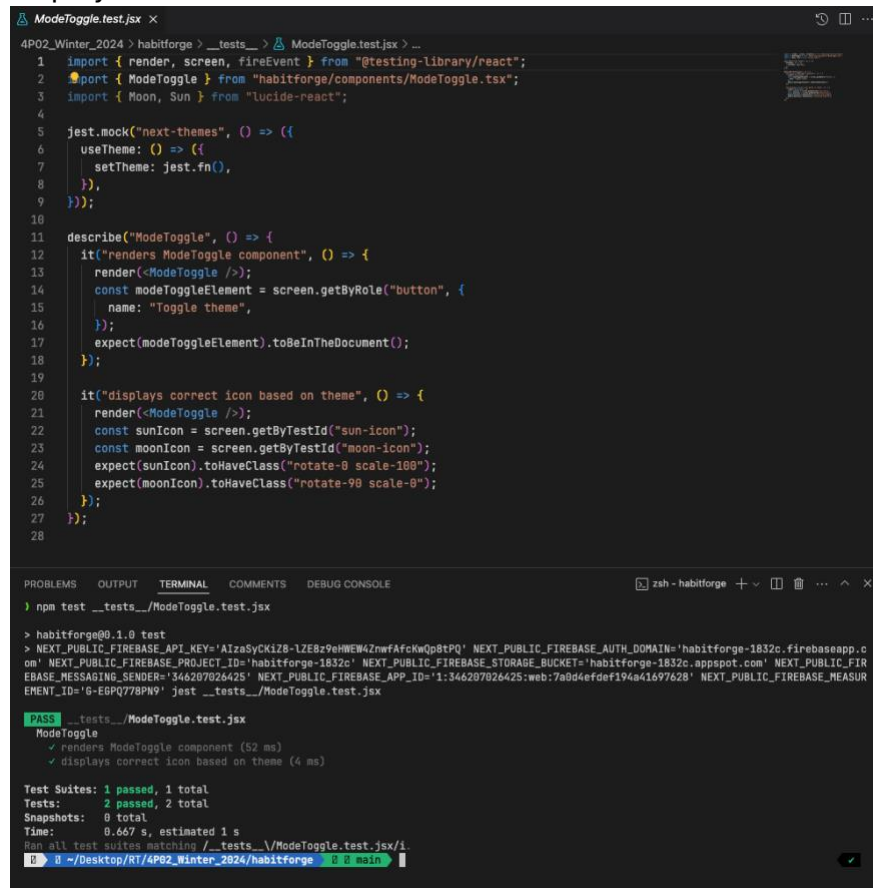
- **Unit Testing - [Test cases branch GitHub](#)**

- Testing was done using Jest, an automated framework for JavaScript.
  - Before implementing Jest, I spent one sprint researching all the different types of testing frameworks such as Vitest, Mocha, and Storybook. However, given our project uses Next.js, TypeScript, and React, the easiest to implement of the three was Jest, as our project already had ESLint, Babel, and Webpack.
- The total test cases that were written was 60, with a total of 12 test suites (See photo below). Link to the test cases:
- I created the test cases such that for each major component that is used in HabitForge, there is a respective test class for that component testing functionality of the component. View them @ [Test cases branch GitHub](#)
- Jest works by creating mock data, and then simulating a scenario by rendering the component.
- In the photo below, there are a total of 12 test classes for each component, with a total of 60 test cases written. Tests are in `__tests__` folder.

```
PASS  __tests__/GroupHabitsDialog.test.jsx
PASS  __tests__/AudioHome.test.jsx
PASS  __tests__/ModeToggle.test.jsx
PASS  __tests__/Podium.test.jsx
PASS  __tests__/AnalyticsSec.test.jsx
PASS  __tests__/MuteProvider.test.jsx
PASS  __tests__/Edit.test.jsx
PASS  __tests__/TimeElapsed.test.jsx
PASS  __tests__/UseSpeechRecognitionHook.test.jsx

Test Suites: 12 passed, 12 total
Tests:      60 passed, 60 total
Snapshots:  0 total
Time:       1.965 s
```

- Below, I'll explain how Jest works by testing functionality of dark/light mode component, which changes the page to light/dark mode. In the photo below, we are running test cases on the **ModeToggle.tsx** component in a test case class called **ModeToggle.test.jsx**. First, using jest, the light/dark mode theme is mocked, and then, two test cases are written. The first checks if the toggle button renders correctly, and the second checks if the correct icon is displayed based off the theme selected.



```
ModeToggle.test.jsx
4P02_Winter_2024 > habitforge > __tests__ > ModeToggle.test.jsx > ...
1 import { render, screen, fireEvent } from "@testing-library/react";
2 import { ModeToggle } from "habitforge/components/ModeToggle.tsx";
3 import { Moon, Sun } from "lucide-react";
4
5 jest.mock("next-themes", () => ({
6   useTheme: () => ({
7     setTheme: jest.fn(),
8   }),
9 }));
10
11 describe("ModeToggle", () => {
12   it("renders ModeToggle component", () => {
13     render(<ModeToggle />);
14     const modeToggleElement = screen.getByRole("button", {
15       name: "Toggle theme",
16     });
17     expect(modeToggleElement).toBeInTheDocument();
18   });
19
20   it("displays correct icon based on theme", () => {
21     render(<ModeToggle />);
22     const sunIcon = screen.getByTestId("sun-icon");
23     const moonIcon = screen.getByTestId("moon-icon");
24     expect(sunIcon).toHaveClass("rotate-0 scale-100");
25     expect(moonIcon).toHaveClass("rotate-90 scale-0");
26   });
27 });
28
```

```
PROBLEMS OUTPUT TERMINAL COMMENTS DEBUG CONSOLE
> npm test __tests__/ModeToggle.test.jsx
> habitforge@0.1.0 test
> NEXT_PUBLIC_FIREBASE_API_KEY='AIzaSyCKi28-1ZEBz9eiHWEW4ZnwAfckwQp8tPQ' NEXT_PUBLIC_FIREBASE_AUTH_DOMAIN='habitforge-1832c.firebaseio.com' NEXT_PUBLIC_FIREBASE_PROJECT_ID='habitforge-1832c' NEXT_PUBLIC_FIREBASE_STORAGE_BUCKET='habitforge-1832c.appspot.com' NEXT_PUBLIC_FIREBASE_MESSAGING_SENDER_ID='346207026425' NEXT_PUBLIC_FIREBASE_APP_ID='1:346207026425:web:7a084ef0ef194a41697628' NEXT_PUBLIC_FIREBASE_MEASUREMENT_ID='G-ESPQ778PN9' jest __tests__/ModeToggle.test.jsx
PASS __tests__/ModeToggle.test.jsx
  ModeToggle
    ✓ renders ModeToggle component (52 ms)
    ✓ displays correct icon based on theme (4 ms)
Test Suites: 1 passed, 1 total
Tests: 2 passed, 2 total
Snapshots: 0 total
Time: 0.667 s, estimated 1 s
Ran all test suites matching /__tests__\/ModeToggle.test.jsx/i
B ~ /Desktop/RT/4P02_Winter_2024/habitforge 0 0 main
```

- There are some components you will notice that don't have test cases or test classes written for them. This is because it would be redundant, as the major components I have tested use those components in the implementation. So, by only testing the major component, it directly tests the functionality of the other components as well.

Overall, I was a Project Manager and a Developer during the lifecycle of HabitForge being developed. As a project manager, I was able to write progress reports, setup project instructions and documentation, maintain sprint notes, facilitate risk management by ensuring we follow proper GitHub branch structure strategies, and making sure all features requested by the product owner are delivered in the final release of our software. As a Developer, I was able to develop the speech-to-text component, allowing a user to create a habit verbally. I also did the unit testing by writing 60 automated test cases using Jest for

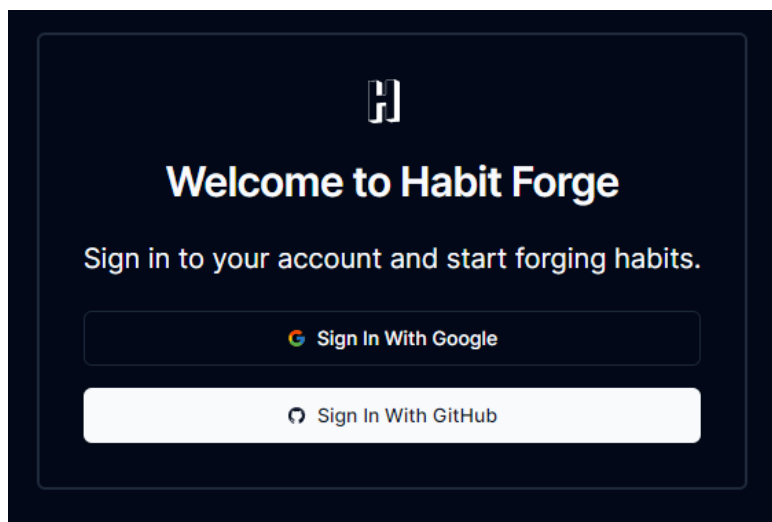
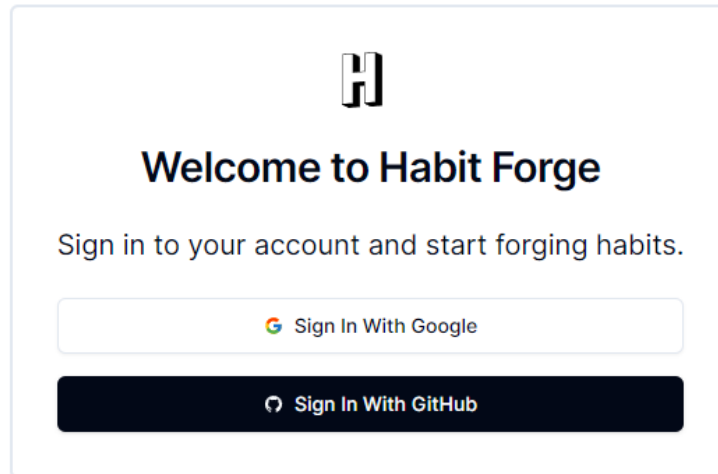
every major component within our project.

## **Rafael [Developer]**

As a Developer for the production team, I played a crucial role in transforming the Product Owner's vision into a functional, user-friendly, and engaging digital experience. My main responsibilities were to research and develop app features, authentication methods, data management and integrity and any other requirements proposed by the Product Owner under the supervision of the Project Manager. My deliverables included:

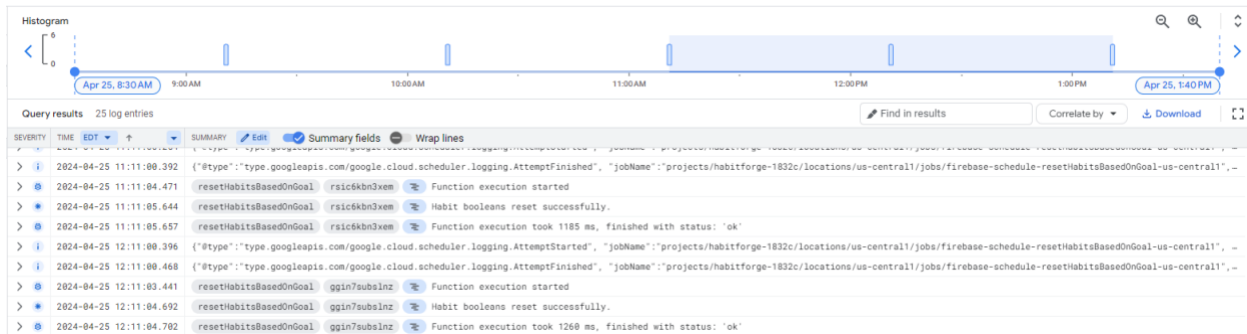
- **Graph Representation**
  - Researched how graphs can be used to represent habit data/tracking
  - Developed a POC via Chart.js and Plotly on how we could benefit from either JavaScript library
- **Sign-in Methods**
  - Add other sign-in methods via firebase authentication
  - Connect authentication providers via a providers developer account, manage client secrets, client ID's and callback URL's. User's can sign-in HabitForge via a:
    - Google account
    - GitHub account
  - Provided the scope and budget for this project, most sign-in methods provided by firebase were not a viable option as most authentication developer accounts required users to pay
  - Integrated sign-in methods with the frontend UI React components for light and dark mode





- **Rank System**
  - Developed a ranking formula for scoring users habit tracking behaviours via habit data
  - Rewarding users for good habit tracking practices
  - The ranking formula goes through each user and each habit as follows:
    - Creating weights for habit variables
      - CompletionRateWeight = 0.6
      - GoalWeight = 0.2
      - FailedWeight = -0.1
      - SkippedWeight = -0.1
      - StreakWeight = 0.5
    - Calculate completion rate by:  
Habits Completed / Total Habits
    - Calculate score for each variable. CompletionScore, GoalScore, SkippedScore, StreakScore
    - Add all scores together for to get the final users score

- Rank each user based off such score to the leaderboard where users can see where they stand among their peers
  - Handle cloud functions errors for users with no habits and other habit data
  - Launch function to run every 24 hours at midnight
- Habit Streaks Functionality**
  - Keeping track of consecutive completed habits for each users' habits
  - Checking if the difference between the last completed date and the current date is equal to or less than 24 hours. Increment streak variable if true or reset to 0 if false
  - Reward players for having longer habit streaks
  - Run cloud function before calculating users ranks
- Manage and debug using Google's Cloud Scheduler**
  - Force run functions for troubleshooting and testing purposes
  - Viewing logs to see errors points



- System Notifications**
  - Assisted Nico with insight on firebase as it is where we did most of our development work from both being to its environment and services

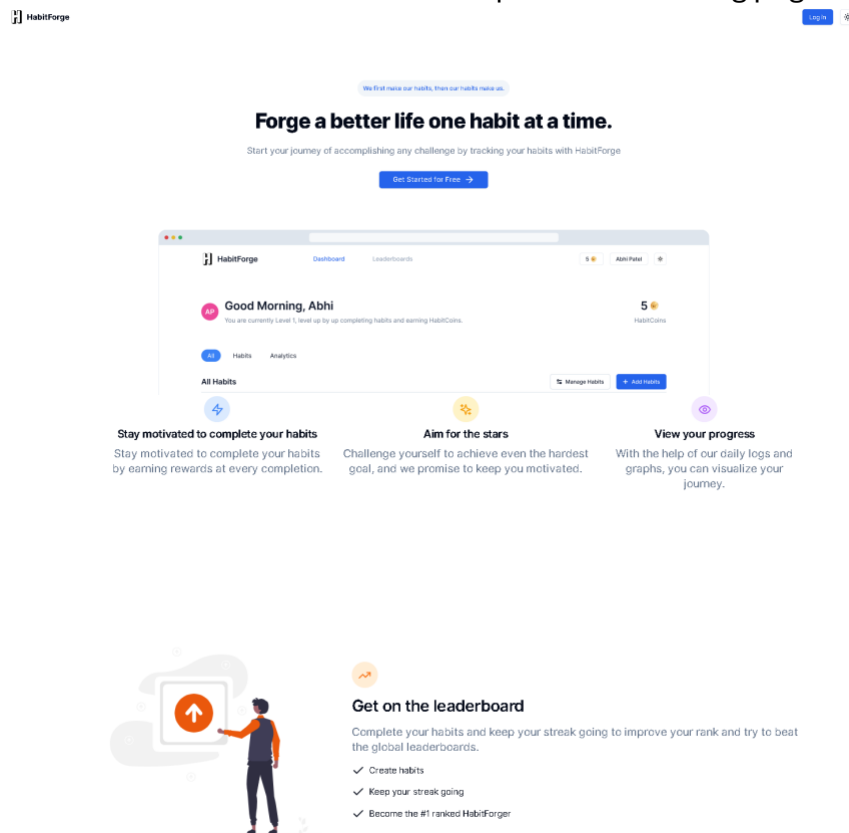
## Zakir Raza

As a developer working with HabitForge, my role focused on researching and functionally developing some parts of the HabitForge web app. To meet the product owner's requirements, work was to be done on a timely manner and meeting project objectives. Throughout the four months' time given to finish the project, it was split up into six sprints where the first sprint was mostly researching while the rest were implementing functions. Working on this project has also allowed me to collectively work together with my team helping each other out to meet sprint deadlines. Additionally, I was also able to gain experience on working with React and TypeScript, thus improving and expanding my coding knowledge.

Here are my contributions:

- Research:**

- During sprint one, my research focused on figuring out how we can limit the number of habits users can have per level. Therefore, we decided to create a habit cooldown function where for each level, players will only have a certain number of habits. To implement this, the level is multiplied by two which equals the number of habits allowed. This functionality is important as it prevents exploitation and promotes fair gameplay among players.
- **Landing Page:**
  - To create the landing page of the app, I worked collectively with my teammate Ahmed. This was done on one machine, therefore one person (me) worked on writing the code, while the other person (Ahmed) helped guiding and detecting issues. Some challenges that we came across was mostly trying to understand react and typescript, but eventually we managed to work it out. Below are some examples of the landing page that we created.



- **Icons:**
  - Another part of the project I worked on was adding more icons for players to choose among based off their habits. Many different people have many habits and it's important for an app like this to have a diverse set of habit icons to choose from.
- **Analytics:**
  - This section of the project was another important feature to implement. Here I developed the front-end of the analytics section which shows users, for example, the number of habits completed, weekly progress, streaks, etc.

Additionally, the section also shows the percentage of the habits or group habits completed. This part was the most challenging part of the project for me to add since I had to create the function that calculates the percentage of habits completed and make a circular percentage path based upon it. The goal was to first use a library that supported creating this, however, since we were using TypeScript, most modules did not support it. Therefore, I had to create my own.

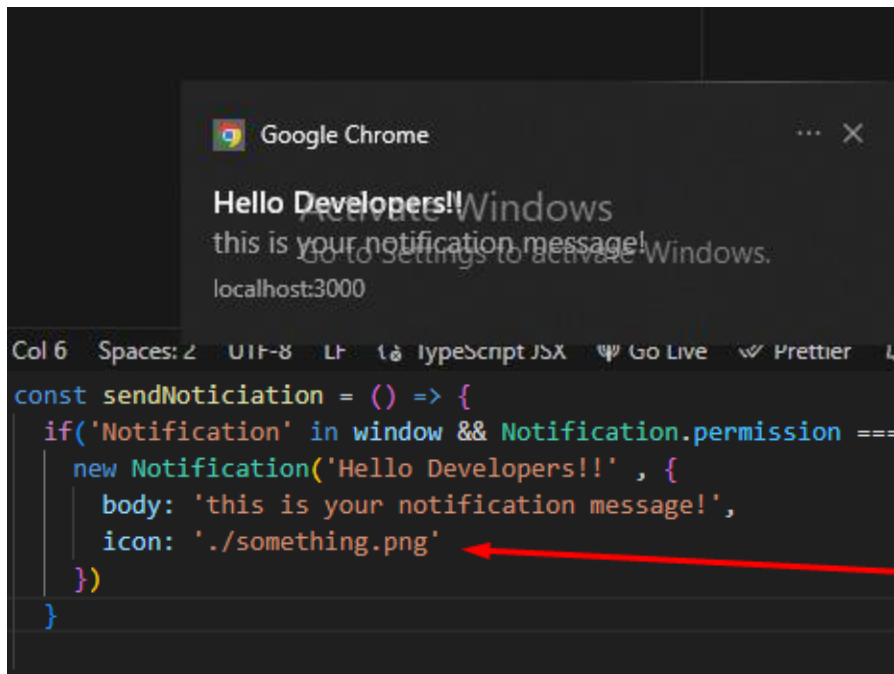


## Nico [Developer]

As a developer at HabitForge, my role was pivotal in making the product owners feedback into a fully functional website by implementing key features and functionalities and helping other developers whenever I can to help them achieve/fix their problems and features. My responsibility was to research mainly about Firebase and how we can get the database set up for certain features as well as research more about how we can make system notifications, Toast Notifications and Test cases which Ameen will have implemented. I was learning Next.js, Shadcn, firebase, Typescript so there was a little bit of a learning curve initially for some things.

These are my contributions:

- **System Notifications:** <https://youtu.be/3HVtQipgf70> <- Demo
  - I did not commit until the feature was fully working, and initially set up Firebase notifications through the database, which will then have a cloud function that runs and also pulls the quotes within that specific Document with the string valued Quotes.
  - It was first tied to a button using a useEffect which was then when pressed it will run the functions that will ask the user for permission and save that and when the notification needed to be sent out, it would have run this function here that took the Notification API class and allowed notifications to be sent along with the image to test at first. This was all changed for what we needed along with our own image.
  - The function that was sending notifications was also set to scheduled updates within the code which would send out daily at 12.

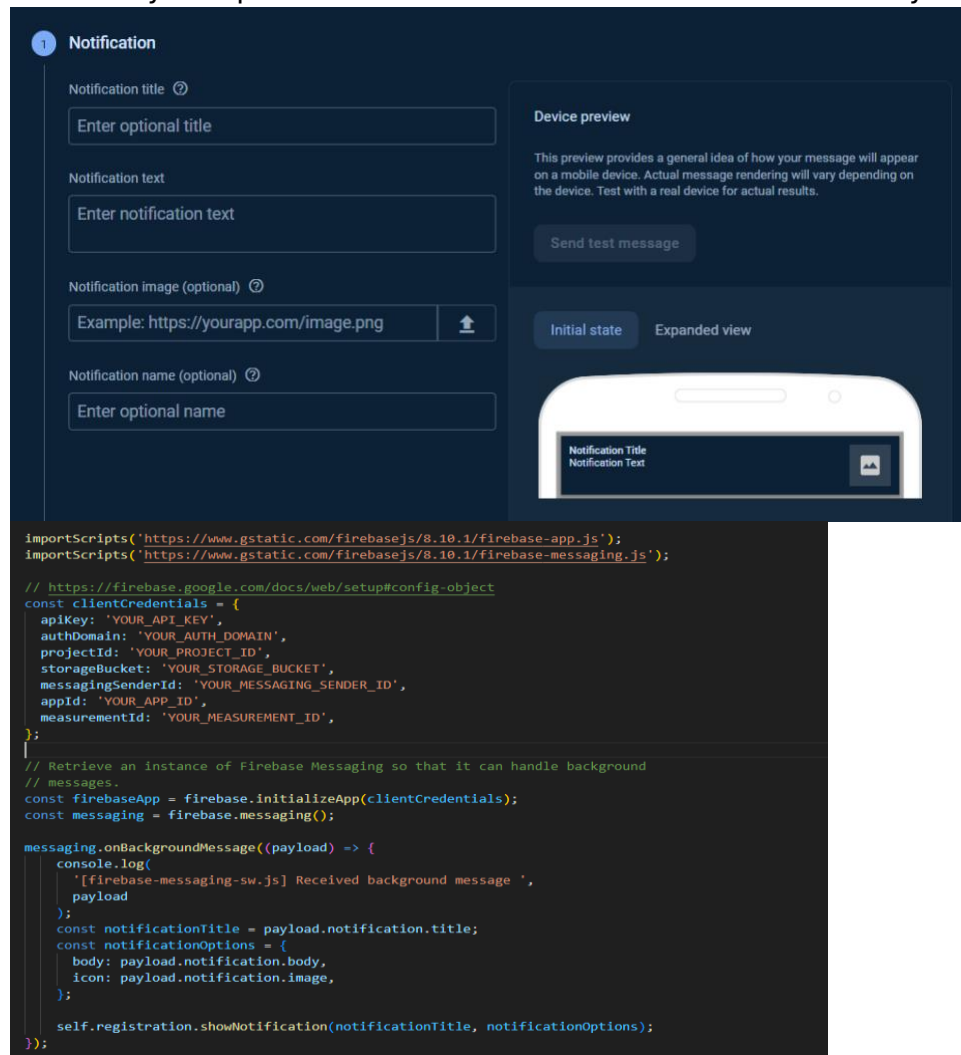


```
import { collection } from 'firebase/firestore';  
  
// Function to fetch data from Firebase and display random quotes as alerts  
export function systemNotification() {  
  const [notifications, loading, error] = useCollectionData(  
    collection(db, 'notifications')  
  );  
  
  useEffect(() => {  
    // Function to display random quote as an alert  
    const displayRandomQuoteAlert = () => {  
      if (!loading && !error && notifications) {  
        const quotes = notifications.map((notification: any) => notification.Quote);  
        const randomQuoteIndex = Math.floor(Math.random() * quotes.length);  
        const randomQuote = quotes[randomQuoteIndex];  
  
        console.log("Random Quote:", randomQuote);  
        // Display the random quote as an alert  
        window.alert(randomQuote);  
      }  
    };  
  
    // Call displayRandomQuoteAlert every few hours (adjust interval as needed)  
    const interval = setInterval(() => {  
      displayRandomQuoteAlert();  
    }, 1000 * 60 * 60 * 2); // Trigger every 2 hours  
  
    // Clear interval when component unmounts  
    return () => clearInterval(interval);  
  }, [loading, error, notifications]);  
}
```

- However, we ended up changing it to a more efficient and automated version which is the Firebase Messaging system, this will allow us to make the notification scheduling automatic and set up within firebase which can access our users directly that sign into our website.
- Setting up and connecting the messaging Firebase System to our client within the code as well as creating the fire-base-messaging file that will then take all the

information from the payload of the client which is linked to our firebase.ts that will then message and get permission to send the notifications if granted.

- It will take the payload.notification.? (body, image) and so on depending on what is written within firebase itself. It is also initialized by our clientCredentials which will have all of information such as, the apiKey, Auth Domain, MessagingSenderId and many more.
- The hardest part about this was firebase, allowing and ensuring that our credentials are already set up and that it is able to access all the database fully as well as users.



The image shows the Firebase Cloud Messaging console interface and a corresponding code snippet. The console is titled "Notification" and includes fields for "Notification title", "Notification text", "Notification image (optional)", and "Notification name (optional)". There is a "Device preview" section on the right that shows a mobile device screen with a notification. Below the console, there is a code snippet in a dark-themed editor showing the initialization of Firebase Messaging and the handling of background messages.









```
importScripts('https://www.gstatic.com/firebasejs/8.10.1/firebase-app.js');
importScripts('https://www.gstatic.com/firebasejs/8.10.1/firebase-messaging.js');

// https://firebase.google.com/docs/web/setup#config-object
const clientCredentials = {
  apiKey: 'YOUR_API_KEY',
  authDomain: 'YOUR_AUTH_DOMAIN',
  projectId: 'YOUR_PROJECT_ID',
  storageBucket: 'YOUR_STORAGE_BUCKET',
  messagingSenderId: 'YOUR_MESSAGING_SENDER_ID',
  appId: 'YOUR_APP_ID',
  measurementId: 'YOUR_MEASUREMENT_ID',
};

// Retrieve an instance of Firebase Messaging so that it can handle background
// messages.
const firebaseApp = firebase.initializeApp(clientCredentials);
const messaging = firebase.messaging();

messaging.onBackgroundMessage((payload) => {
  console.log(
    '[firebase-messaging-sw.js] Received background message ',
    payload
  );
  const notificationTitle = payload.notification.title;
  const notificationOptions = {
    body: payload.notification.body,
    icon: payload.notification.image,
  };

  self.registration.showNotification(notificationTitle, notificationOptions);
});
```

	testing test new	Apr 22, 2024 2:37:19 PM	-	Completed
	HabitForge Setting goals is the first step in turning the invisible into the visible.	Apr 26, 2024 12:00:00 PM	-	Scheduled
	HabitForge Failure is only postponed success as long as courage 'coaches' ambition. The habit of persistence is the habit of victory.	Apr 27, 2024 12:00:00 PM	-	Scheduled
	People with goals succeed because they know where they're going. People with goals succeed because they know where they're going.	Apr 26, 2024 12:00:00 PM	-	Scheduled
	HabitForge The big secret in life is that there is no big secret. Whatever your goal, you can get there if you're willing to work!	Apr 4, 2024 12:00:00 PM	-	Active
	HabitForge Good things happen to those who hustle	Apr 3, 2024 12:00:00 PM	-	Active
	HabitForge Our daily decisions and habits have a huge impact upon both our levels of happiness and success.	Apr 2, 2024 12:00:00 PM	-	Active
	HabitForge If you're walking down the right path and you're willing to keep walking, eventually you'll make progress	Apr 1, 2024 12:00:00 PM	-	Active

### Rank System/Levels:

- Provided Rafael some insight and research about game ranking systems.
- How points could be accumulated
- Firebase -> provided some insight and help for the getting and setting firebase fields for users rank and habit data.

### Firestore:

- A significant portion of my time was dedicated to Firestore, mainly due to the fact that the initial challenges with setting up the cloud functions to interact seamlessly with the database. I focused mainly on configuring firestore to ensure that it can retrieve data and manipulate it within our website.
- One of the key tasks for System Notifications was creating new collections within the database dedicated mainly to notifications where the implementation will retrieve the data stored within the randomized named documents.
- It was my first time being exposed to Firestore, there was a lot of research needed to make sure that I understood what was happening and trouble shooting issues effectively through videos and or documentation provided from Firestore.

**Presentation**

- Prepared slides for 1-25 of the presentation based on the presentation document provided by Ahmed.
- Collaborated with Ahmed to refine and enhance the content for each slide making sure it is clear and concise.



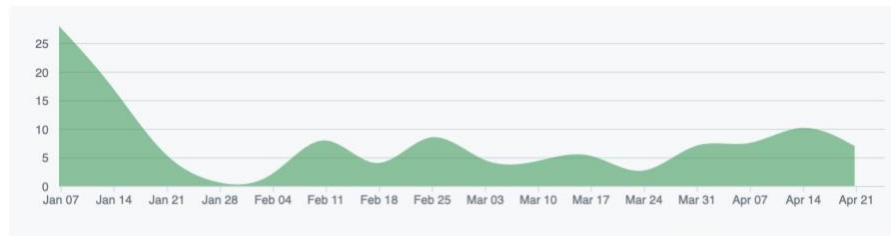
## GitHub Contribution

Here is a screenshot of our GitHub contributions. We had a total of 165 Commits and 22 branches.

Jan 7, 2024 – Apr 25, 2024

Contributions: Commits ▾

Contributions to main, excluding merge commits



## Agile Process

We followed the agile process as it gave us the most flexibility while also keeping everyone on track and have a plan that can be adjusted based on the requirements. We set out a sprint plan with 6 sprints over the 4 months. Each sprint was 2 weeks long and we met every week for checking each others progress and answer any questions or problems that we may have occurred. Every 2 weeks we did a sprint overview, went over the tasks that

were completed and whatever that wasn't completed was moved on to the next sprint. Other than UI designs this never happened as all development tasks were completed on the time.

## **Conclusion**

In conclusion, we as the team of HabitForge are we proud of the work we have been able to do in the 4 months. We have all worked beautifully as a team. We did not have any troubles or problems arise in terms of teamwork or the tasks handed to us. Everyone did the job they were given and put their best foot forward. We also appreciate all the help and feedback that Brendan and our professor Naser.

## **Future**

HabitForge can have a great future if any of us decide to keep working on it. Since the codebase is very adaptable, it is very future proof. None of us have any plans right now for the future but it's always an option when any ideas come up.