SOCIAL FAMILY
Cloud Application Development: Team E

John Philip Wilson, Anton Okhotnikov, Yixuan Xu, Nunzio Gatti, Richa Ranjan, Susmita Gangopadhyay, and Junming Zhang

University of Southampton, MSc Data Science

January 6, 2018

Application URL-
Github-

1 Introduction

With the advent of internet and it's growing popularity over the last decade, children aged between 5-13 have been exposed to a plethora of messaging applications regularly. For children of a young age chatting with friends via a text based chat app can be great way to improve their literacy and build social relationships with friends. However current chat apps target adults and provide little in the way of protection for children of this vulnerable age. Social Family aims to give parent a chat application they can give to there young children which gives the parents control over who there child is able to chat with and allow them to monitor what conversations they are having. It is really important for the parents to know how these applications work, and how old their child should be, to be able to use them, to help avoid *cyberbullying* and other unpleasant conversations. Our target is 5 years and above at the point where children start to read and write. Our objective is to provide young children safe access to text communication with friends and family with the aim of helping them to develop their language and social skills.

2 Prototype functionality

The Social Family app is designed for parents to be able to add another child's contact, and thus authorize the communication between their child and this added contact. When a child comes up with a contact name/ID to be added to his friend list, the request has to go through the Parent. In this way, the parent is aware of who their child is making friends, or having a conversation with. The application functionality is summarized in the following steps:

1. In the first step, when a Parent tries to load the application, they would be redirected to the Google login page, as this app uses the Google login authentication.
2. For the first login, Registration is required. For this, the user is asked to create a 4-digit PIN. This gives the parent the ability to lock the contact list so that no new contacts can be added.
3. The parent, once successfully logged in, would now be able to log in on the child's device, and search for the child's friend ID.
4. Once the parent adds a friend to the child's contact, the child will now be authorized to have a chat session with the friend.When passing to the child the parent locks the application using a pin. The child is not authorized to add new contacts on his own. Parental authentication is a must for this.
5. To send/receive messages, **Twilio** SDK packages are used. Twilio enables the parent to track their child's conversations by keeping the HTTP cookies saved. This way, the parent would be able to monitor the content of the messages as well.

3 TOOLS AND TECHNIQUE

We are aware that chat applications are not new, what was missing was simply the ability to give parents control and support on multiple platforms like web, iOS, Android and Kindle Fire. To this end we aimed to focus on

- Multiple platform support
- Parental control over who your child is talking to.
- Monitoring of conversations

3.1 Third-party Libraries and Cloud Services

- GAE for platform hosting
- MongoDB/Datastore for database
- Google authentication, means one less pass word for the user to remember
- Twilio - A hosted third part chat service, manages chat channels and chat history
- IBM Watson - A hosted third part natural language processing for sentiment analysis on messages to enable automatic parental alerts for concerning messages
- React - A JavaScript library for building user interface
- Material UI - A React components library that implement Google's Material Design,
- Flask - A Python web application framework

3.2 Development tools and techniques

- Github for source code management using feature branching
- Github for Document sharing
- Trello board and agile development approach.
- Visual Studio Code as text editor and IDE
- Create-react-app - An official scaffolding for react
- Webpack - A static module bundler for modern JavaScript application
- Babel - A JavaScript compiler for next generation JavaScript
- Npm for JavaScript dependencies management and production code optimization
- ESlint for JavaScript development.
- Prettier for JavaScipt code formatting.
- React Dev Tool - rowser extension for react development (FireFox and Chrome)
- Pip for Python dependencies management
- Pylint for Python development.
- Autopep8 for Python code formatting.
- We used a code review process before features were merged into the master branch or made live. This helped us stabilize features before merging into the main product. We also setup the repository so that anyone submitting a pull request could not review or merge it,which prevented unchecked code entering the main branch.

4 Relevant Statistics

Total commits to master-142

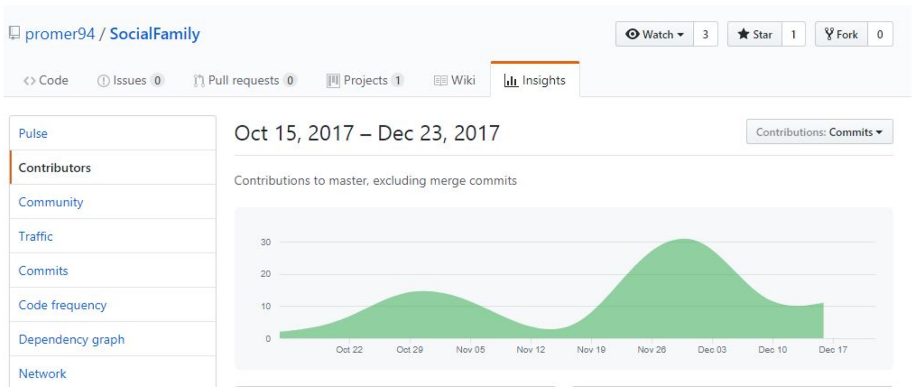

Figure 1: Total number of commits over a period of time

Total lines of code-
Total number of spaces-
Total number of comments-

5 Design and Implementation

Initial design specifications for the prototype was to build a messaging service, following the classic chat app model such as WhatsApp or Skype. We aimed at building separate chat accounts for parents, as well as the child. After few weeks of development, we concluded that this was beyond the scope within the given time frame. So, we came up with a model where there is a single chat channel created and locked by the parent. The child would then be unable to add contacts on his own. The first step towards this was to choose an existing sdk, rather than building a messaging service from scratch, so that we could add or modify the additional functionalities based on our requirements. Twillio was our choice. After successful implementation of twillio sdk,the front end was created using React. Twillio provided storage of chat history and we used MongoDB to store user and other details.
There are four pages in our app.They are

- Signin:The signin page of the app allows users to signin using google credentials.
- Register:The register page allows the user to create an account in our app and set a lock pin . Lock/unlock is only set by parents and child users have not access to this functionality
- Home: This is the main app page with contacts and chat window

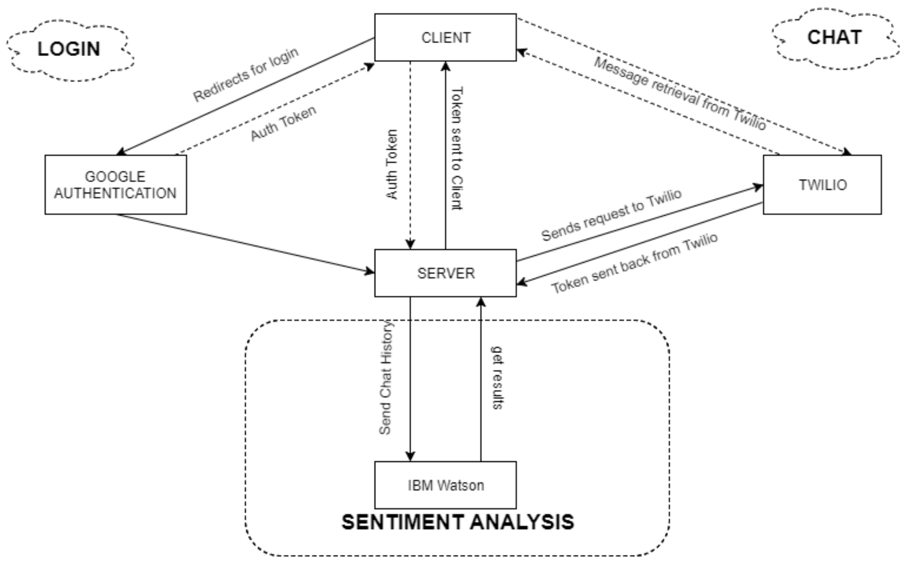

Figure 2: Flow Diagram of the working system

6 Critical Evaluation

6.1 Strength

Scalability-As described in the previous sections, we have used modern and design patterns to keep our code clean, maintainable and testable.Our architecture is aimed to be very scalable as it should be for any cloud application.
Compatibility-The application uses Babel for the javascript code. This enables our application to run on multiple modern browsers like Google Chrome, Firefox and Safari.
Extensibility- Additional features can be added to our app, as the code structure is easy to modify
sentiment analysis-We also made use of a hosted sentiment analysis service by IBM Watson which gave us the ability to automatically highlight messages that might be of concern
quick page refresh-Using React for our front end user interface allowed us to create a rich user experience that was able to refresh the display client side without needing for slow page refreshes

6.2 Challenges

We had technical difficulties in setting up Google Datastore and instead made use of an AWS hosted instance of a mongoDB. If we had more time we would want to replace our database with Google Datastore as it is able to provide the functionality we require and would simplify redeployment of the application.

6.3 Market Research

We have also done a lot of background research and market analysis to find out what exactly we want to build and why. We found that although there are a lot of similar apps in the market,none of them serves the same purpose as ours.Here is a list of apps that we found

- Monster messenger:
  URL-https://monster-messenger.com/
  Available platform-ios,android
  Target Audience-under 13
  Not supported on Amazon kindle,No sentimental analysis
- Roo kids:
  URL-http://www.rookidsapp.com/
  Available platform-ios, android ,amazon kindle
  No specific target audience No sentimental analysis
- MMguardian:
  URL-http://www.mmguardian.com/welcome-to-messenger-kids/
  Available platform-android iphone
  No specific target audience
- Facebook messenger kids:
  URL-https://www.wired.com/story/facebook-for-6-year-olds-welcome-to-messenger-kids/
  Available platform-ios, android ,amazon kindle
  Target Audience-Target Audience-under 13 No sentimental analysis

What makes us unique is that we have an app which targets pre-teens(5-13) which supports all the platforms, especially amazon kindle with sentimental analysis.