SE 3XA3: Software Requirements Specification SocialPy

Team 1, SocialPy
Anando Zaman, zamana11
Graeme Woods, woodsg1
Yuvraj Randhawa, randhawy
Due February 12, 2021
Tag SRS-Rev.1

April 8, 2021

Contents

List of Tables

List of Figures

This document describes the requirements for SocialPy. The template for the Software Requirements Specification (SRS) is a subset of the Volere template

1 Project Drivers

1.1 The Purpose of the Project

Social media user adoption has increased drastically over the past few years as the world becomes further digitally connected. Communication between people from different parts of the world has traditionally been conducted through phone calls. However, this no longer provides the same personal "touch" and often feels sub-par. The purpose of this project is to re-implement an open-source social media application with extended features that create a better and more personal user experience. This app will allow users from all over the world to communicate via short text posts such as about their current status or location-based information. Additionally, the app will feature account authentication to ensure the integrity of the users posts.

1.2 The Stakeholders

1.2.1 The Client

The clients for this project include Dr. Ashghar Bokhari (Professor of SE 3XA3 course) along with the TA for this course (Maryam Hosseinkord, ~~Justeen Randev, Shardool Patel, etc~~).

1.2.2 The Customers

The customers for this project will be any person or public/private entity that is seeking ways to improve communication amongst each other and the general public. This can include everyday people (general public), Media outlets such as news channels, and firms who want to further increase their presence for advertising online. The general public will use this for general communication while news channels will be using it for sharing headlines from around the globe. Advertisement firms can use it to posts ads to entice users to buy their products.

Table 1: Revision History

Date	**Developer(s)**	**Change**
February 4, 2021	Anando Zaman	Copy template.
February 5, 2021	Anando Zaman	Worked on Project Drivers section and beginning of functional reqs section and some use cases.
February 9, 2021	Anando Zaman	Completed the Project Issues section. Made progress on some of the NFRs section. Completed Project Drivers Section
February 11, 2021	Yuvraj Randhawa	Completed the Functional requirements and use case sections.
February 11, 2021	Graeme Woods	NFRs section & Off-the-Shelf Solutions

1.2.3 Other Stakeholders

N/A

1.3 Mandated Constraints

1.3.1 Solution Constraints

Description: The product shall be written in the Python 3 programming language.
Rationale: All developers are familiar with Python 3 and there is an existing API available to communicate with the Firebase database.
Fit criterion: The software is written in Python 3.

Description: The product shall function only when connected to the internet.
Rationale: The product actively retrieves data via the Firebase Database API and as such, requires a stable internet connection.
Fit criterion: The product will display an error if no internet connection detected.

1.3.2 Implementation Environment of the current system

The application will be cloned from the gitlab repo and executed via the IDE of the user's choice. Examples include PyCharm, VS Code, Python IDLE. The *main.py* file will be executed in order to interact with the program.

1.3.3 Partner or Collaborative Applications

The product will use the Firebase Cloud Database and Firebase API to store and retrieve user account data.

1.3.4 Off-the-shelf software

The following off-the-shelf software will be utilized:

- Firebase API for Python (Available here).
- Python 3 (Available here)
- PyCharm IDE (available here)

1.3.5 Anticipated Workplace Environment

The product can be used on any desktop machine that is capable of executing Python Scripts and has an internet connection.

1.3.6 Enterprise Constraints

The product will be available for use to any user or organization that has Python3 and the Firebase API dependencies installed.

1.3.7 Budget Constraints

Budget constraints do not apply to this project as it is free for a small set of users. However in the long-term, a license or database credit has to be purchased from Firebase if the number of API write requests exceeds 20K/day or read requests exceed 50K/day. The quota and pricing is a pay-as-you go model and more information can be found here, regarding infrastrucure costs.

1.3.8 Schedule Constraints

The project deliverables must be completed by their respective deadlines. The upcoming deadlines include:

- Proof of Concept (POC) - February 21, 2021
- Test Plan, Revision 0 - March 5, 2021
- Design & Documentation, Revision 0 - March 18, 2021
- Revision 0 Demonstration - March 22, 2021
- Final Demonstration - April 5, 2021
- Final Documentation - April 12, 2021

1.3.9 Naming Conventions and Terminology

Table 2: Naming Conventions and Terminology

Term	**Definition**
Product	Social-Media-Terminal App
OS	Operating system
FireBase	Cloud Database provider
User	The person using the product
API	Application Programming Interface
Python	Computer Programming Language
Repo	Repository of information
GitLab	A type of Repo for Version control
BE	Business Event
App	Short for application
Clone	Copy the contents from a repository to local storage

1.4 Relevant Facts and Assumptions

It is assumed that the user will have a stable internet connection and is able to comprehend the English language and Alphabet. The original project has roughly 1100 lines of code. It is assumed that the user has a desktop with at least a keyboard or mouse to interface with the application and that their hardware is capable of running Python 3.

2 Functional Requirements

2.1 The Scope of the Work and the Product

SocialPy is a terminal-based social media application developed using the Python programming language. This application will facilitate the process of communication of several users in a format similar to a online thread/forum. Users post their current status/emotions/thoughts and any other content in the form of short 3 to 4 sentence messages. These messages will be visible under the users "activity wall" which other users will be able to view. Additionally, users will be able to follow other users and view the content they post. The expected outcome is that people will be further connected and have a form of communication that is more enjoyable.

2.1.1 The Context of the Work

This product is designed to be used by the general public and large organizations such as advertisers and news outlets. This is in an effort to communicate their content across to hundreds of thousands of people that are all using the application. In the original code of the product, there was a lack of user security as anyone could post messages on behalf of someone else simply by using their username. There was no user authentication involved which resulted in significant repudiation and data integrity issues. In addition, the lack of data persistence drastically reduced functionality of the application as the user data activity would be erased once the application was closed. The goal of SocialPy is to solve

these shortcomings and develop a more refined product that ensures data integrity and persists the data so that it does not get erased. In addition, several new features like delete post and view profile will also be added to further enrich the user experience.

2.1.2 Work Partitioning

1. User attempts to login/register
2. User attempts to post information.
3. User attempts to view list of recent posts.
4. User attempts to follow another user
5. User attempts to remove a post
6. User attempts find another user

2.1.3 Individual Product Use Cases

Use case 1
Title: Registering an account to the server
Trigger: User wants to create an account to use the app.
Pre-condition: User has a stable internet connection.
Scenario/outcome:

- System asks for username and password credentials
- User inputs the information
- System sends the credentials to the Firebase Authentication server to validate information.
- System returns with PASS or FAIL state for authentication.
- If PASS state, system transitions to homescreen.
- If FAIL state, system re-prompts the user.

Use case 2
Title: Create user session (login)
Trigger: User wants to login and utilize the app.
Pre-condition: User has registered an account to the server and has a stable internet connection
Scenario/outcome:

- System asks for username and password credentials
- User inputs the information
- System sends the credentials to the Firebase Authentication server to validate information
- System returns with PASS or FAIL state for authentication

- If PASS state, system transitions to homescreen
- if FAIL state, system re-prompts the user to input correct username or password

Use case 3
Title: End user session (logout)
Trigger: User wants to end their active session
Pre-condition: User has created a session and has a stable internet connection
Scenario/outcome:

- User inputs "LOGOUT" into the system
- System ends the session the user has created and returns them to the starting screen

Use case 4
Title: User view posts
Trigger: User wants to view either their or another user's posts
Pre-condition: User must create a session and have a stable internet connection
Scenario/outcome:

- User inputs "POSTS VIEW" followed by the username of the profile that they want to view posts of
- System sends the username to the Firebase server to verify whether the username is registered
- If the username is registered, the posts will be displayed along with the identification number of each post
- If the username is not registered, the system will re-prompt the user to input a registered username.

Use case 5
Title: User add posts
Trigger: User wants to add new posts
Pre-condition: User must create a session and have a stable internet connection
Scenario/outcome:

- User inputs "POSTS ADD" followed by the content of the post
- The system will send the content of the post to the Firebase server
- The post will be created and given a unique identification number

Use case 6
Title: User updates posts
Trigger: User wants to update existing posts
Pre-condition: User must create a session, have at least one existing post and have a stable internet connection
Scenario/outcome:

- User inputs "POSTS UPDATE" followed by the identification number of the post and the new content the post will contain

- The system will send the identification number of the post to the Firebase server
- The server will verify whether there exists a post created by the current user with the identification number specified
- If there does not exist such a post, the system will inform the user that no such post either exists, or is registered to the current user
- If there does exist such a post, the content will be modified with the content inputted by the user

Use case 7
Title: User deletes post
Trigger: User wants to remove an existing post
Pre-condition: User must create a session, have at least one existing post and have a stable internet connection
Scenario/outcome:

- User inputs "POST DELETE" followed by the identification number of the post
- The system will send the identification number of the post to the Firebase server
- The server will verify that there exists a post created by the current user with the identification number specified
- If there does not exist such a post, the system will inform the user that no such post is registered to the current user
- If there does exist such a post, the post will be deleted

Use case 8
Title: Follow User
Trigger: User wants to follow another user
Pre-condition: User must create a session and have a stable internet connection
Scenario/outcome:

- User inputs "PROFILE FOLLOW" followed by the username of the profile they want to follow
- The system will send the username to the Firebase server
- The server will verify that a profile with the specified username exists
- If there exists such a profile, the user will now be following the user that they inputted
- If there does not exist such a profile, the system will inform the user that no such user exists

Use case 9
Title: View User Profile
Trigger: User wants to view profile of another user
Pre-condition: User must create a session and have a stable internet connection
Scenario/outcome:

- User inputs “PROFILE VIEW” followed by the username of the profile they want to find
- The system will send the username to the Firebase server
- The server will verify that a profile with the specified username exists
- If there does not exist such a profile, the user will be informed that no such user exists.
- If there exists such a profile, the system will display the= information(location, name, followers) of the user that they searched for.

Use case 10
Title: Edit Profile
Trigger:
Pre-condition: User must create a session and have a stable internet connection
Scenario/outcome:

- User inputs “PROFILE EDIT”
- The system allows the user to change their full name
- The system sends the updated information to the Firebase server

2.2 Functional Requirements

FR1. The system shall be able to retrieve user data via the FireBase API.

- Rationale: The data from Firebase is what is being used to conduct actions such as authentication, posting, viewing posts.

FR2. The system shall be able to authenticate the user

- Rationale: Authentication is required so ensure data integrity and to prevent unauthorized access to an account.

FR3. The system shall be able to register new users.

- Rationale: New users must have an account in order to interact with the system, otherwise it would be difficult to identify who made a post.

FR4. The system shall be able to retrieve posts by username

- Rationale: The user may want to view the posts by other users. Each username is associated with a profile and posts, thus information must be retrievable via username.

FR5. The system shall be able to delete posts by the author

- Rationale: The user may change their mind and want to delete a post at anytime, thus ability to delete posts is necessary.

FR6. The system shall be able to reset the users password

- Rationale: In case the user forgets their password, the system will provide a way to reset it via email verification.

FR7. The system shall be able to allow users to follow other users

- Rationale: The user must be able to follow others so that they know who to get their favourite content from.

FR8. The system shall be able to allow users to make posts

- Rationale: Since this is a social media app, each user will likely want to make some posts. As such, the system MUST have this ability. Otherwise, the application becomes useless.

FR9. The system shall be able to allow users to update their posts

- Rationale: The user may want to edit their posts at anytime if they changed their mind about something they posted earlier.

FR10. The system shall have short messages to indicate what specific commands do.

- Rationale: The user will need to know what specific commands are in order to use the application. Otherwise, it becomes difficult if user were to guess and check.

FR11. The system shall allow users to view profile information(name, location, followers) about other users.

- Rationale: The user must be able to view this information, otherwise they will not know who they are following or who to follow.

FR12. The system shall allow users to edit their profile location.

- Rationale: The user may change their location and want to ensure the most up-to-date information is available for others to view.

FR13. The system shall allow users to edit their name.

- Rationale: The user may change want to change their name or hide it with something else just because they want to.

FR14. The system shall allow users to delete their account.

- Rationale: The user may want to stop using social media altogether and do not want an online prescence on the platform.

FR15. The system shall allow users to delete followers.

- Rationale: The user may change their mind about following someone.

FR16. The system shall allow users to find/view followers of a given user.

- Rationale: The user may just be curious or want to find mutual friends that someone else follows.

3 Non-functional Requirements

3.1 Look and Feel Requirements

3.1.1 Appearance Requirements

NFR1. The system shall follow a command-line text based UI.

3.1.2 Style Requirements

NFR2. The system shall use short and descriptive labels.

NFR3. The system shall display black and white text.

3.2 Usability and Humanity Requirements

3.2.1 Ease of Use Requirements

NFR4. The system shall be used by any person ages 13 and over. Fit Criterion: if 80% of users said they are 13 years old or older and had neutral or positive experience in regards to difficulty ~~The system shall have short messages to indicate what specific commands do.~~

NFR5. The system shall be easy to navigate and use such that the user is able to their desired action in under 15 seconds.

3.2.2 Personalization and Internationalization Requirements

NFR6. The system shall have all text written using the English alphabet and language.

3.3 Performance Requirements

3.3.1 Speed Requirements

NFR7. The system shall query posts and profiles within 2 seconds.

NFR8. The system shall be able to authenticate the user within 4 seconds.

3.3.2 Precision or Accuracy Requirements

NFR9. The system shall be able to retrieve/update timestamp of posts data in YY/MM/DD HH:MM:SS format up to 2 decimal places.

NFR10. ~~The system shall allow a maximum of 200 characters for a given post.~~

3.3.3 Reliability and Availability Requirements

NFR11. The system shall ~~have~~ be available with 99% up-time. The remaining 1% will be used for maintenance and updates.

3.3.4 Scalability or Extensibility Requirements

NFR12. The system shall be able to handle user interactions ~~thousands of users concurrently~~ by at minimum five users concurrently.

NFR13. The database shall be able to increase its storage memory if greater number of users and posts activity occur.

3.4 Operational and Environmental Requirements

3.4.1 Expected Physical Environement

NFR14. The system shall be operated from a desktop or laptop using Windows, Mac OS, Linux, or any other system that supports Python.

NFR15. The system shall be in an environment that has internet access.

3.4.2 Requirements for interfacing with Adjacent Systems

NFR16. The system shall be able to communicate with the Firebase Database.

3.4.3 Installability requirements

NFR17. The system shall not need to be installed on the users' machine.

NFR18. The system shall be a simple "clone" of the Gitlab repository.

NFR19. The system shall be executable via Python in under 2 seconds.

3.4.4 Release requirements

NFR20. The product shall be released on Gitlab repo with updates every month based on user feedback and feature requests.

3.5 Maintainability and Support Requirements

3.5.1 Maintenance Requirements

NFR21. Our entire team will be required to make changes and improvements over the lifespan of the product.

NFR22. Bug fixes will need to be made as issues are found through customers using the software.

NFR23. Our project will likely have improvements and new features that must also be implemented to stay compatible with new devices. This will be essential to product maintenance.

3.5.2 Supportability Requirements

NFR24. Our software will have a 'help' command that can be run to show the user a list of commands that can be executed.

NFR25. This 'help' command will be run automatically after the user opens the software for the first time and will act as the tutorial.

3.5.3 Longevity Requirements

NFR26. The project will have a long lifespan and because it doesn't use any 3rd party API's (other than our Firebase database) there is a low likelihood of a external change breaking our application.

NFR27. As computers evolve, there may be a future OS update that will be incompatible with our software.

3.6 Security Requirements

3.6.1 Access and Confidentiality Requirements

NFR28. Passwords associated with a username shall be hashed and unavailable from the general public to ensure confidentiality.

NFR29. Only users with a registered account shall be able to view posts.

3.6.2 Integrity Requirements

NFR30. The system shall only allow authenticated users to post via their unique immutable username.

NFR31. The system shall timeout if more than 5 failed login attempts have been made consecutively.

3.7 Cultural Requirements

NFR32. The system shall use Canadian English spelling.

3.8 Legal Requirements

NFR33. The system shall not violate any copyright terms.

NFR34. The system shall prohibit all forms of offensive content.

NFR35. The system shall allow freedom of speech unless it is related to offensive content.

3.9 Health and Safety Requirements

NFR36. The system shall remove/deactivate users who promote cyber-bullying and organized crime ~~or other actions of that sort~~.

4 Project Issues

4.1 Open Issues

4.1.1 Python3 and Firebase future support

As the Python language continously evolves with incremental updates year over year, it is unclear whether the current version of Python (Python 3) would be discontinued. Discontinuation of the current version could cause harm to the product as it can result in outdated libraries that no longer function as expected. For example, the Firebase API library is designed for Python 3. As new versions of Python and Firebase release in the future, there is no gurantee the same functionality or methods would still work. Since Firebase is a cloud platform, the application uses a dedicated Firebase API to communicate with the database in order to retrieve the data. If Firebase undergoes changes, it can impact the application as it may cause some friction when trying to work with deprecated code.

4.2 Off-the-Shelf Solutions

4.2.1 Existing APIs

Firebase API is the only off-the-shelf solution used in order to store and extract data from the application.

4.2.2 Existing Products

Since the project is based on re-implementing an existing social media code base, therefore some of the codebase is very similar except that it is now ported into Python. This project could also be compared to social media websites like Twitter, especially when it was in it's infancy. Our product will be much simpler than Twitter is now, however the main goals of allowing users to share text posts is the same.

4.3 New Problems

4.3.1 Copyright issues

Must ensure that the game does not infringe copyright compared to similar features of other social media applications.

4.3.2 Effects in the Current Environment

The application database is hosted on Firebases' cloud platform. However, increased/overloaded activity or usage requests can exceed quotas and potentially reduce performance.

4.3.3 Potential User Problems

The application is all displayed through a terminal window which can cause eye strain when reading large bodies of text. In addition, the application can potentially cause extended physical health problems when using a computer including but not limited to, carpal tunnel syndrome, blurred computer vision, and musculoskeletal problems.

4.3.4 Limitations in the Anticipated Implementation Environment That May Inhibit the New Product

Since the application uses a database hosted from Firebase Cloud Platform, it is possible that the daily data request quotas could exceed depending on the number of users active at a given time. This can drastically increase maintenance costs.

4.4 Tasks

The steps to deliver the product with in-depth details of each stage of the development process is outlined in the Gantt Chart of this project. These Gantt charts are updated on a periodic basis and cover all tasks, deliverables, and content breakdowns that are to be completed by the given deadlines. The development phases are outlined and specify the team members that are responsible for a given task.

4.5 Migration to the New Product

N/A.

4.6 Risks

The risks to this project are very low if any at all. Security is a low concern as authentication is handled by the Firebase back-end system. However, the lack of two-step authentication can potentially cause security issues if password is compromised. ~~Additionally,there may be risks associated with the application performance and response time if several users use the product at once which could send clusters of requests to the back-end. This is a low risk as the cloud database is able to scale with the number of users. Poorly optimized code during development may cause some issues on the user-experience side as users often have little patience when using applications and may switch to a competing product.~~This is a difficult task to mitigate as Firebase does not provide a two-step authentication feature.

4.7 Costs

The only cost the application will have is that of hosting the database to the third-party cloud database provider, Firebase. For the purposes of development, this will be free-to-use but will need to be upgraded to a pay-as-you-go model or one of the enterprise plans provided from the Firebase website.

4.8 User Documentation and Training

Explicit user training or instructions will not be necessary. The application will display to the user detailed prompts in order to do tasks such as create posts, view posts, etc. As such, no explicit documentation is required for the user to review prior to using the application.

4.9 Waiting Room

N/A

4.10 Ideas for Solutions

The original source code is in Java and does not use any specific APIs for persisting the user data. Our version will be in Python and use Firebase as the database to persist the user data(ie; posts, profile information, etc). The database will generate a unique userID(UID) for each user that registers an account. The program will retrieve this only when the user has been authenticated. Through this, the user's interactions such as "view posts" and "add posts" will be conducted. Therefore, each transaction will be applied against the users' UID in order to maintain and update their activity throughout the application.

References

[1] Volere Requirements Specification Template Edition 13 - August 2007. James and Suzanne Robertson, Atlantic Systems Guild, 2007.

5 Appendix

N/A

5.1 Symbolic Parameters

N/A