BankChain - Product Planning

Chainable Technologies

H.G. van de Kuilen

rvandekuilen, 4226151

J.C. Kuijpers

jckuijpers, 4209915

E.J. Sennema

esennema, 4496582

M.R. Kok

mrkok, 4437659

I. Dijcks

idijcks, 4371151

May 2017

Contents

1 Introduction

Planning is important when doing a project, especially when working with a team, and with intermediate deadlines. Plannings gives insight in what is possible in a timeframe, and what is not. It also indicates when the project is behind a schedule.

This document focuses on the planning of the BankChain project of Chainable Technologies. It contains an explanation about the product, with a high level backlog explaining the features. Next, a roadmap is given for the duration of the project. Finally, a definition of a finished product is given.

2 Product

We will build an Android application that enables the user to verify that a public key belongs to a certain IBAN. The goal is to create more trust between two peers and possibly, verifiably, exchange their names. This is achieved by sending small amounts of money to each others bank accounts with an encrypted challenge and response. The product is a part of a larger project in the same context: to create a web of trust on Android phones.

2.1 High level backlog

There are a few elements of our application that we cannot do without. In this Chapter we will briefly discuss these elements. The elements are in order of priority.

Generate a private and public key

Our app has to generate a private and public key pair, to be able to encrypt and decrypt the messages send to a users bank account.

Banking API

Our app has to be able to interact with the API of a bank (Bunq in our case), to be able to transfer money and send messages between bank accounts.

Challenge Response

Our app has to be able to send a challenge to another users bank account, and reply with a response that is a valid solution to the proposed challenge.

Output to Web of Trust

Once you have verified that a public key belongs to a bank account, the app has to store that result in the Web of Trust. That way the user doesn't have to repeat the verification process. The user will see this in the app as a list of public keys that are trusted.

Intuitive UI

As IBAN numbers and public keys can become quite confusing numbers for the user, we will have to build an intuitive UI where the user has to deal with this as little as possible.

QR code

Since typing a public key is very error prone, the user should be able to scan a QR code to input the public key it wants to connect to an IBAN.

Integration with other teams

For the context, preferably, our product would be merged with two other groups, creating a larger app that is an autonomous, working, web-of-trust.

2.2 Roadmap

In sprint 7 and 8 we will have to integrate our app with the app of another project group. We will continue to discuss the integration with the other groups throughout the development.

Sprint 1

- Write the Architecture Design.
- Write the Product Planning.
- Write the Product Vision.

Sprint 2

- Implement a basic challenge/response system.
- Bank API integration (Bunq).
- Build a basic GUI.

Sprint 3

- Discuss integration with other project groups.
- Further improve our challenge/response system.

Sprint 4

- Create a Blockchain Interface.
- Improvement to the GUI.

Sprint 5

- Implement QR code scanning for exchanging public keys

Sprint 6

- Output the trusted public keys to the Web of Trust
- Show a list of trusted keys on the users request.

Sprint 7

- Integrate our app with other project groups.

Sprint 8

- Integrate our app with other project groups.

3 Product Backlog

We use MoSCoW to sort our user stories.

"The MoSCoW method is an acronym commonly used to represent four hierarchical priority groups. Each requirement within a group shares the same priority." (Hatton, 2008). According to Hatton, the groups are as following: Must have: The minimum functionality that the agent should have. Should have: Advanced functionalities that would be nice to have if possible. Could have: Advanced functionalities that would be nice to have if possible, but a bit lower prioritisation than Should have. Won't have: Extended functionality that the agent would not have, as these will not be implemented in the current software.

3.1 User Stories of features

Must have
The next features are essential.

3.2 User stories of defects

Because we haven't started developing our product, we don't have user stories of defects yet.

3.3 User stories of technical improvements

Because we haven't started developing our product, we don't have user stories of technical improvements yet.

3.4 User stories of know-how acquisition

Some user stories don't revolve around adding new features, but rather performing research as to what is the best way to implement such a feature. As we develop the product, we will probably have to adapt, and add more of these user stores. Following is a list we are currently planning to do.

- Find out the best way to store a public key in a QR code.
- Research if other banks have useable API's.
- Compare the various ways to generate public and private keys.

3.5 Initial release plan

4 Definition of done

Finally we have to define when our work is done. Earlier in this document we defined when our final product is done, so in this section we will focus on when backlog items and sprints are done.

A user story is done when it is implemented in our app, the code is tested, and approved by other developers. Tests are for example Unit tests, or manual testing. We aim for 80% branch coverage. The code also has to live up to our documentation standard. When all tests pass, en there are no more bugs in the code, we can merge the user story into our working version.

Since a sprint is a specific time period, the sprint is done at the end of the week. If not all user stories from the sprint are implemented, they will have to be carried over to the next sprint. The work of a sprint is done when the application contains all features that were planned for that sprint. We have to run all the tests that we did for testing the features, with the addition of user tests to confirm the product works properly. We will also confirm with the client that we build the features in our sprint the way he wanted.

The final product is finished when it meets the following requirements.

- All features from our roadmap are implemented.
- All unit- and user tests pass.
- The client is happy with out product.
- The code is properly documented.
- We pass the final SIG test.

Only when we meet all these requirements, we can consider our product to be done.

5 Glossary

Glossary

API Application Programming Interface. 3

blockchain is "a distributed database that maintains a continuously growing list of records, called blocks, secured from tampering and revision" (Wikipedia, 2017). 5

GUI Graphical User Interface. 5

IBAN International Bank Account Number. 3, 4

References

Hatton, S. (2008). Choosing the right prioritisation method. In *Software engineering, 2008. aswec 2008. 19th australian conference on* (pp. 517–526).

Wikipedia. (2017). *Blockchain — wikipedia, the free encyclopedia.* Retrieved from `https://en.wikipedia.org/w/index.php?title=Blockchain&oldid=778628009` ([Online; accessed 4-May-2017])