GitGroup - A Web Based Software Development Management System

Runbo ZHAO
Jingyu BAO
Xuying CAO
Chengxiang YIN

Advisor: Edmund S. Yu
Course: CSE 682 Software Engineering

October 27, 2018

Contents

Chapter 1

Introduction

1.1 Purpose

The document is an official statement of what the system developers should implement. It includes both the user requirements for a system and a detailed specification of the system requirements. The document is essential for outside contractor who developing the software system. It is also useful to write a short supporting document that defines the business and dependability requirement for the system. The document is for system customers, managers, system engineers, system test engineers, system maintenance engineers.

1.2 Product Scope

Gitgroup design for people who has less development experience such as project manager, project members and people who are related to the project. It is an easy access online application that served for product management team based on Github API.
Our online application sets up by several software development tools. In langrage part, we use both TypeScript and JavaScript. TypeScript is an open-source programming language.It is a strict syntactical superset of JavaScript, and adds optional static typing to the language.(?????????????)Javascript is a language which is also characterized as dynamic, weakly typed, prototype-based and multi-paradigm.????For frontend and backend, we used ReadctJS(Framework), Bulma(CSS Framework), Frontawesome(Icon Set and Toolkit) and NodeJS(JS Engine), TS-Node(TypeScript execution and REPL for node.js), ExpressJS(Framework), Mongoose(ODM) in different part. In the test process, our tool is Jest. It is a Javascript testing software.(???test??????????test??????????)
Also we regard MongoDB as our host DBMS, which is one of the most popular database management tool. Others like Git, Heroku, VSCode, Postman, are also included in developing process.

1.3 Definitions, Acronyms and Abbreviations

Owner	Who set up the project or mainly charge of the project
Collaborator	Who assists owner or project participants
Project	An element in GitGroup, including multiple repository
Repository	
Issue	
KanBan	
Cards	

Table 1.1: Definitions, Acronyms and Abbreviations

1.4 Overview of The Document

The rest of the document will include overall description of Gitgroup in following perspectives. Product perspective, product functions, user characteristics, operating environment, design and implementation constraints, assumptions and dependencies. For interface requirement, we put efforts on user interfaces, hardware interfaces, software interfaces and communications interfaces. we will also include requirements like user requirements, system requirements, functional requirements, non-functional requirements and system models like use case diagram, activity diagram, sequence diagram, class diagram, system architecture, web design.

Chapter 2

Overall Description

This section of the SRS will include general factors that affect the product and its requirements. This section does not state specific requirements. Instead, it provides a background for those requirements, which are defined in detail in Section 3 of the SRS, and makes them easier to understand.

2.1 Product Perspective

The background of the Gitgroup coming from project managers are often busy to managing multiple projects at the same time. Sometimes, they are not familiar with details with products especially in technology perspective. However, Git and Github have a high entrance standard which project manager hard to manipulate it. For example, when you want to create an issue to suggest a new idea or track a bug, you need to learn markdown syntax, emoji code. And it is even more complex when you do pull request and merge the project. Comparing with Git and Github, GitGroup is an easier access tool . People who has little development experience, such as product manager. They also can handle GitGroup easily. The only thing they need to do is sign up an account and access to the GitHub api. The user can create online group chat, schedual and resord conference. After finding project idea, our online app also provide the task management service. In each project, every user will have their own grade. Furthermore, the online web can have a recommendation system which help the product manager and programmer match each other. In conclusion, it is a online service based on the GitHub API, and improve the user experience.

2.2 Product Functions

Our product focus on easy access, clearly, convenience, efficiency. Fuctions are surrounded our goals. Mainly function lists in the following.

1. Online meeting: Online meeting group chat. Record the conference. Make conference scheduling form and alarming the coming meeting.

2. Task management: The function is Like GitHub task management, but Gitgroup has better UI design. Make TODO list also included in task management, users create an issue to suggest a new idea or track a bug by means of setting up a TODO list. Then user can organize and assign tasks to their team members.

3. Developer grading system: Record the working of any team member. According the performance of users in all team which he joined, get users a grade.

4. Developer finding system: If you are a team leader, you can find a perfect developer who has related skill to your project. If you are a software developer, you can post your information to find suitable project for you.

2.3 User Characteristic

A high-level diagrammatic representation of the user activity is depicted below

2.3.1 For Owner

2.3.2 For Collaborator

2.4 Operating Environment

2.5 Design And Implementation Constraints

2.6 Assumptions And Dependencies

2.6.1 Assumptions

2.6.2 Dependencies

Chapter 3

Interface Requirements

3.1 User Interfaces

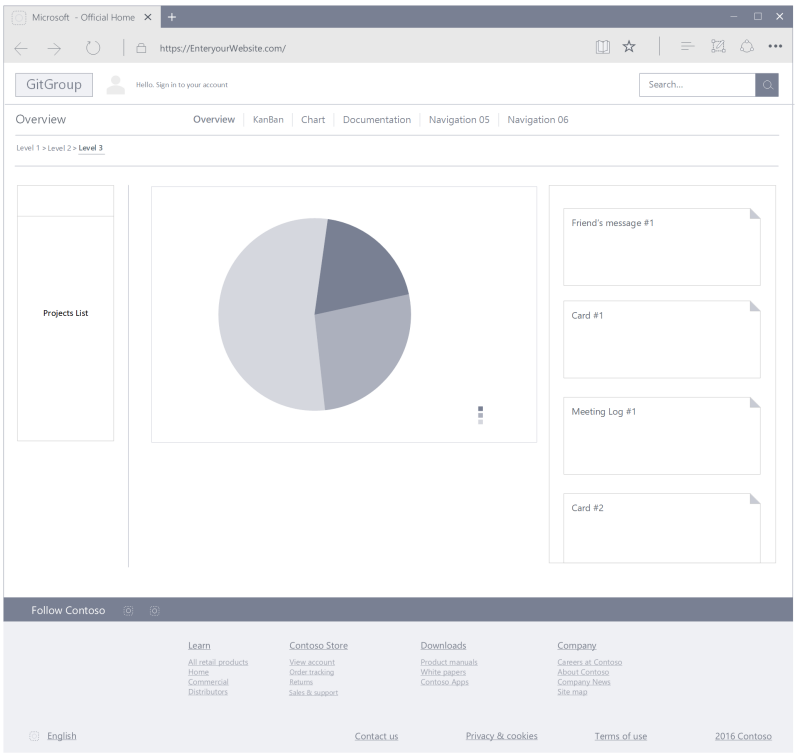

Figure 3.1: Overview User Interface (Details in Appendix A)

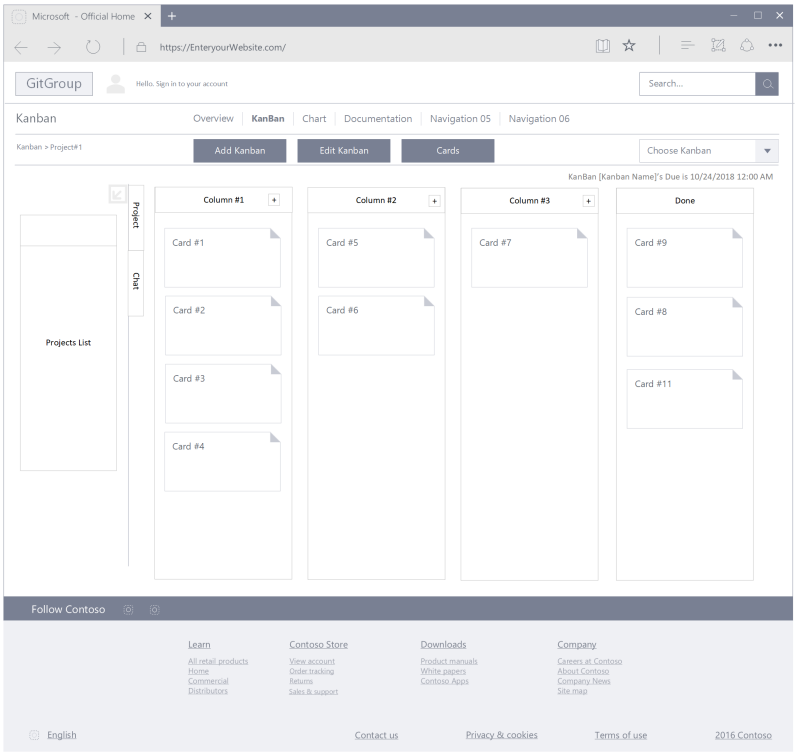

Figure 3.2: KanBan User Interface (Details in Appendix A)

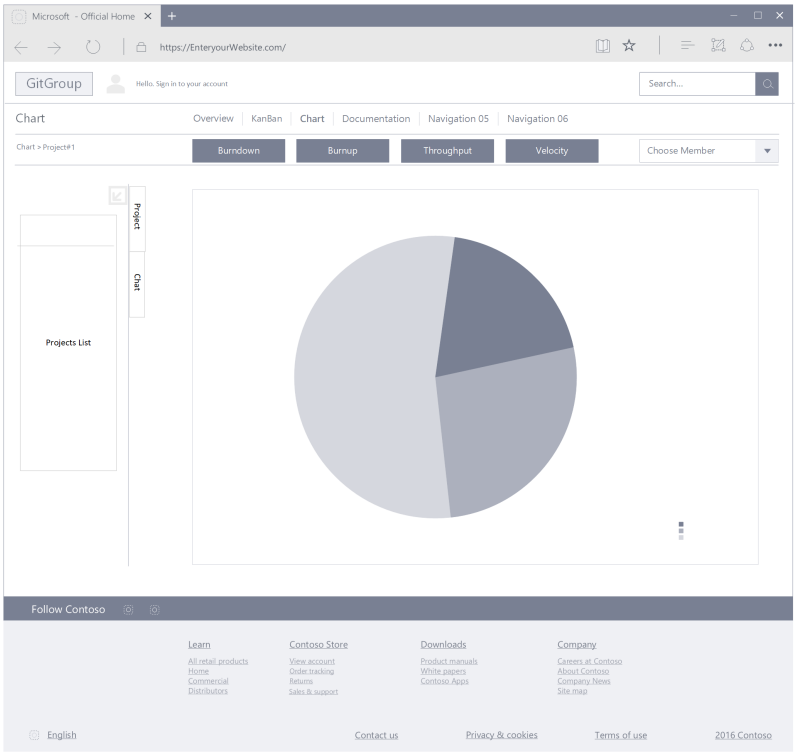

Figure 3.3: Chart User Interface (Details in Appendix A)

3.2 Hardware Interfaces

3.3 Software Interfaces

3.4 Communication Interfaces

Chapter 4

Requirements

4.1 User Requirements

4.1.1 Functional User Requirement

1. User shall be able to create a new project and new team in GitGroup.

2. User shall be able to manage GitHub repositories and team members like add or remove repositories and members.

3. User shall be able to use Kanban board to organize issues by dragging issue cards among several columns representing different stage of the development processes.

4. User shall be able to analysis the development duration, quality and the team work situation of the project by using a variety of agile charts and analytics. also manage tasks like suggesting a new idea or tracking a bug.

5. User shall be able to organize an online meeting group chat, schedule and record the conference within the team or the global scope.

4.1.2 Non-Functional User Requirement

1. The application shall be able to make GitHub easy to use, especially friendly for the person who has little development experience such as product manager.

2. The application shall be quickly restored to operational status after a failure occurs.

3. The app shall be reliable to uses with no downtime.

4. The application shall be able to provide maximum security against malicious attack.

4.2 System Requirements

4.2.1 Functional System Requirement

1.1 Developers can create a new project and make a team with other developers.

1.2 Developers can remove a project and collaborators of the project after checking if the project is empty.

2.1 Developers can manage their team by inviting to or removing collaborators from their projects.

2.2 Developers can manage their repositories by adding or removing repositories of their projects.

3.1 Developers can classify issues by different stages of development process via putting them in different columns of Kanban board.

3.2 Developers can add or remove a Kanban from their Kanban board.

3.3 Developers can customize their own stage by editing column name except Done column.

3.4 Developers can close issues by dragging it to the Done column, which will automatically close issues in the GitHub.

4.1 Project manager can track and communicate the progress of their projects by burn down and burn up charts.

4.2 Project manager can measure how much work a team can used in eXtreme Programming and Scrum from throughput chart.

4.3 Project manager can analysis the velocity of project going from velocity chart.

5.1 Team leader can organize an online meeting group chat within their team.

5.2 Developers can organize an online meeting group chat within the global.

5.3 Developers can receive a meeting notification from team leader.

5.4 Developers can set an alarm for notifying the coming events on a conference schedule form.

5.5 Developers shall be able to record content of meeting on a meeting notebook.

4.2.2 Non-Functional System Requirement

1.1 The user shall be able to use all the app functions without any kind of training. The average number of questions call about how to use app shall not exceed 10 per day.

1.2 The app shall be available for any kind of mobile device and PC.

2.1

3.1 The app shall be available to all users during whole day (Mon-Sun, 00:00 00:00)

4.1 The app shall not expose contact information to other users.

4.2 The app should minimize the amount of personally identifying information (PII) that it collects.

Chapter 5

System Models

5.1 Use Case Diagram

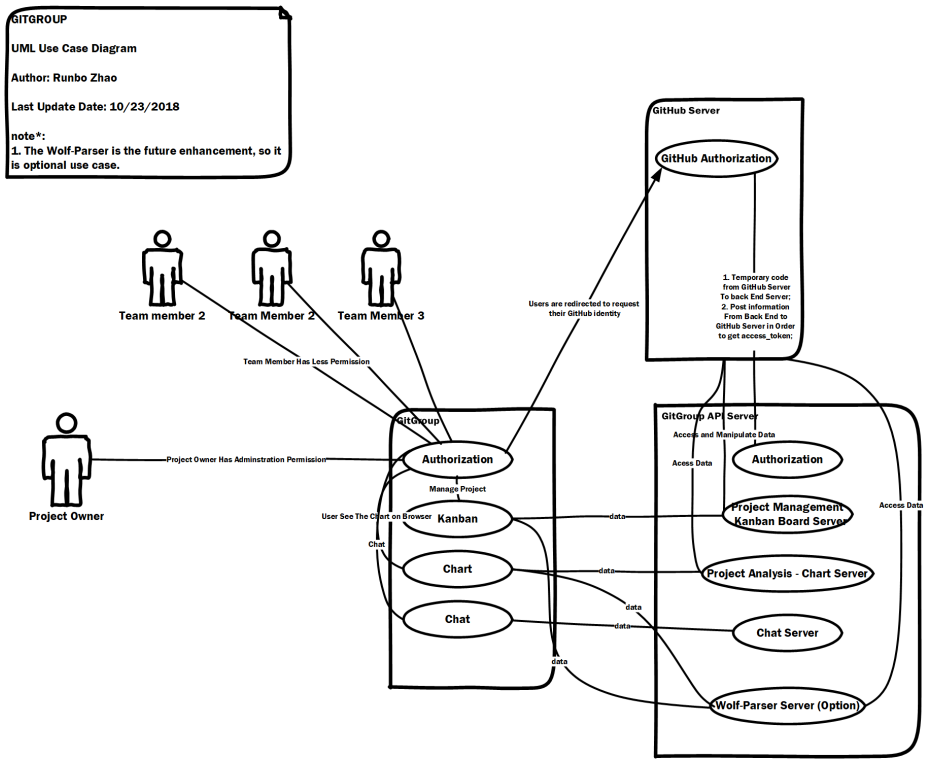

Figure 5.1: Use Case Diagram (Details in Appendix A)

5.2 Activity Diagram

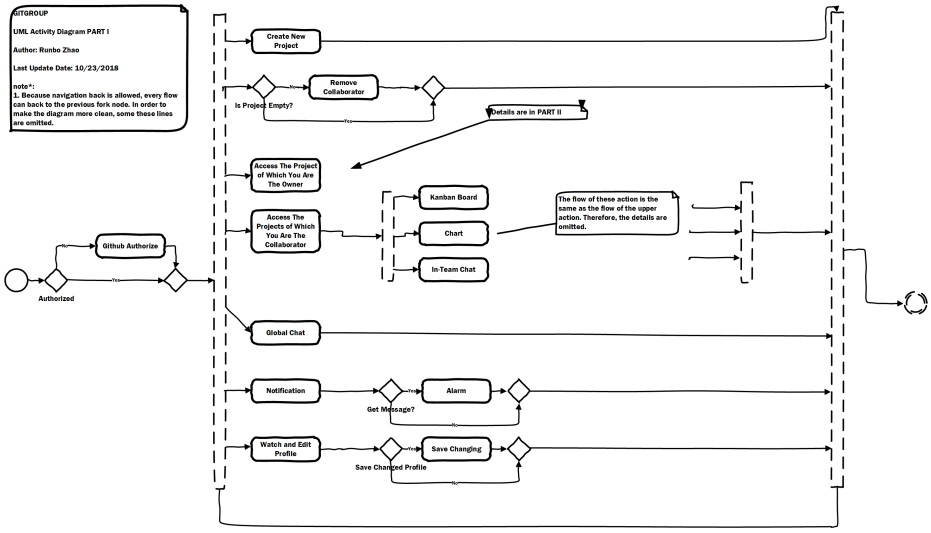

Figure 5.2: Activity Diagram PART I (Details in Appendix A)

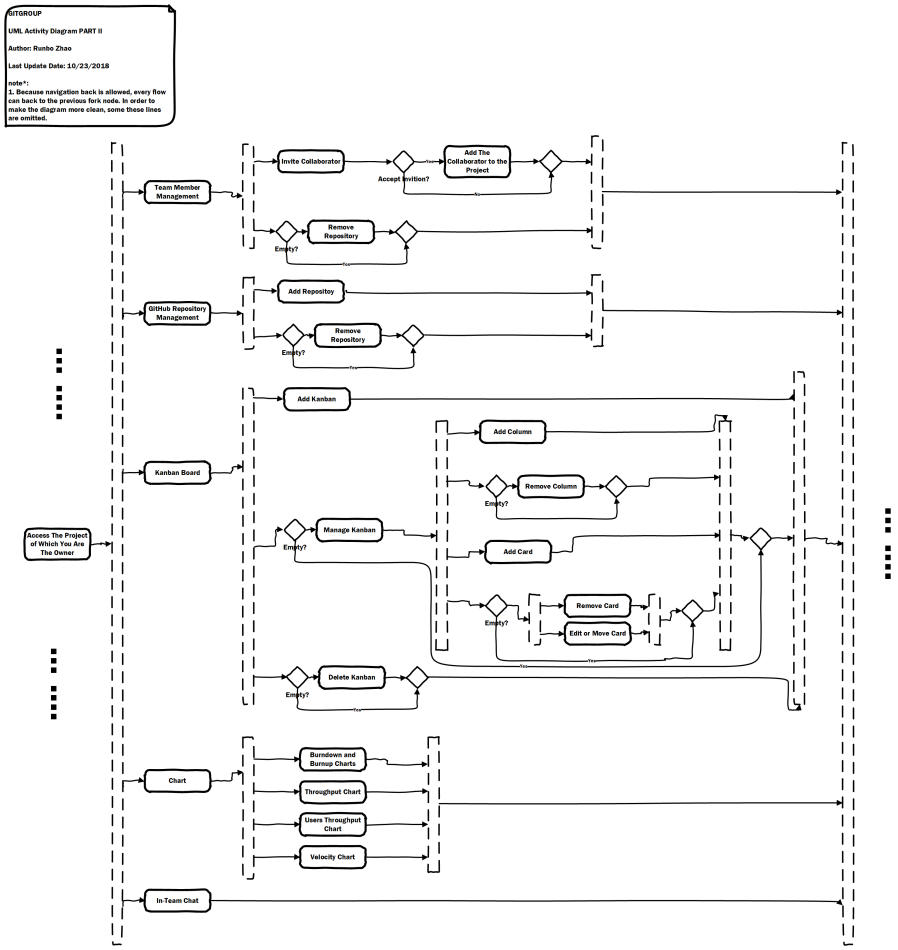

Figure 5.3: Activity Diagram PART II (Details in Appendix A)

5.3 Sequence Diagram

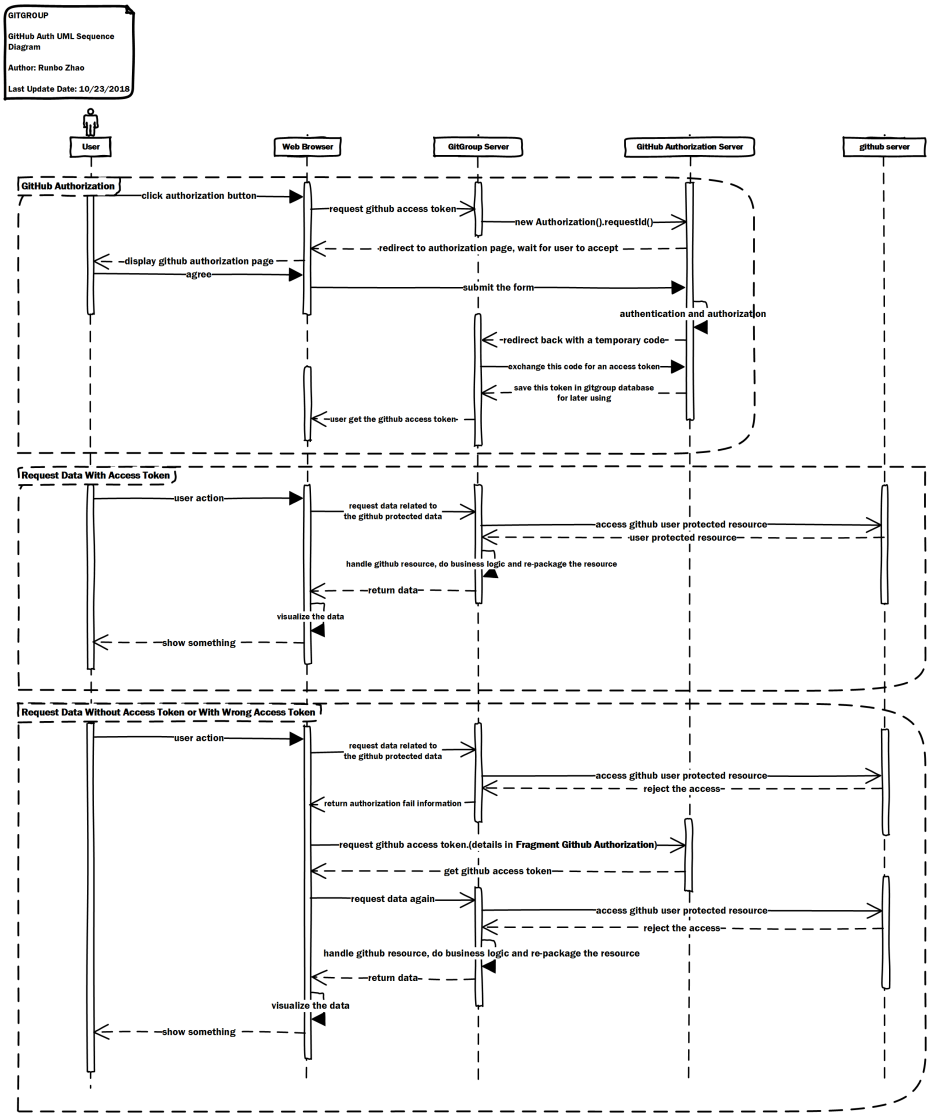

Figure 5.4: Sequence Diagram For Authorization (Details in Appendix A)

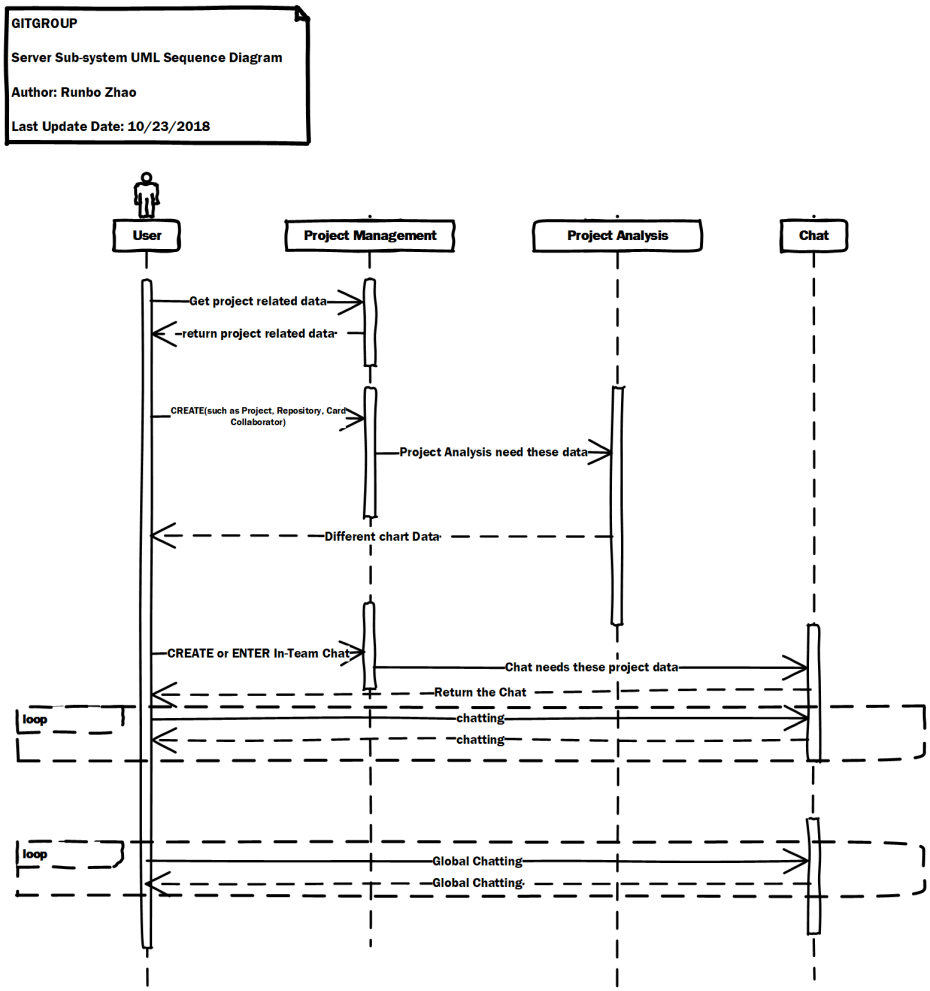

Figure 5.5: Sequence Diagram For Sub-Systems (Details in Appendix A)

5.4 Class Diagram

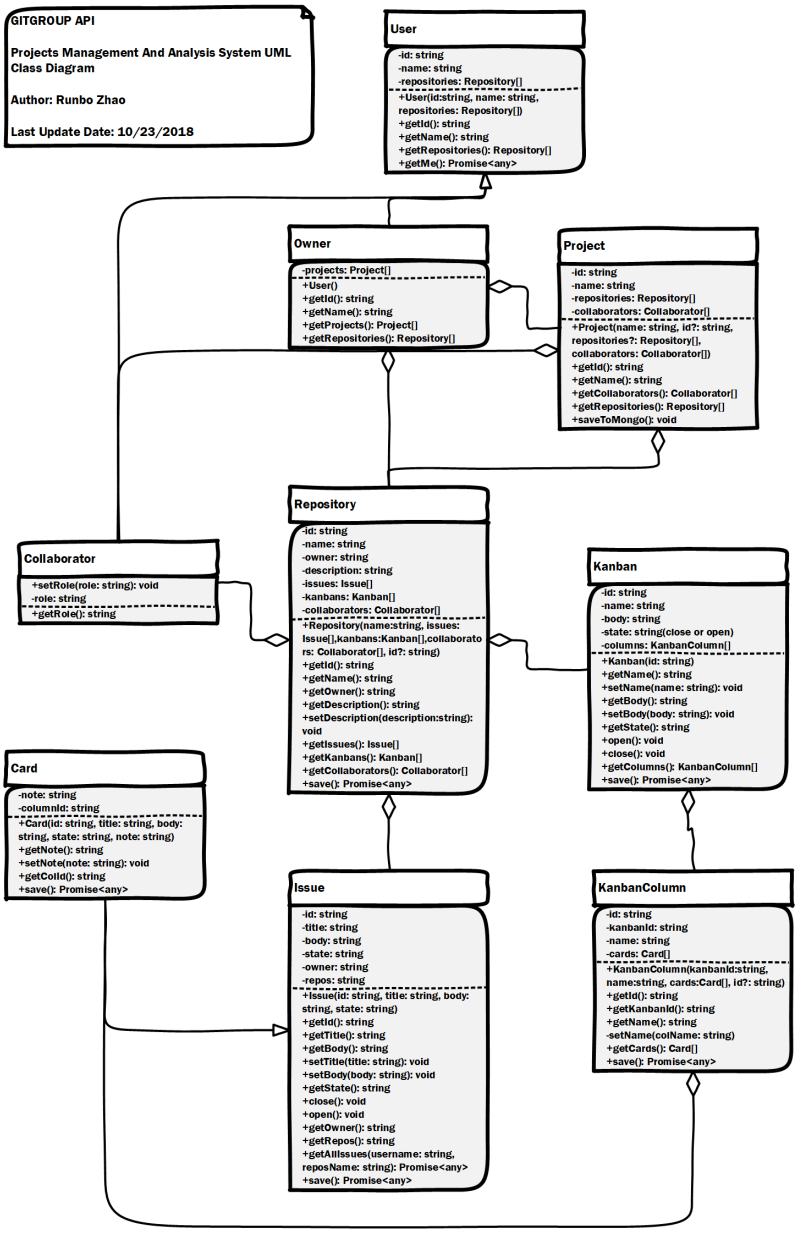

Figure 5.6: UML Class Diagram (Details in Appendix A)

Chapter 6

System Architecture

Appendix A

Appendix

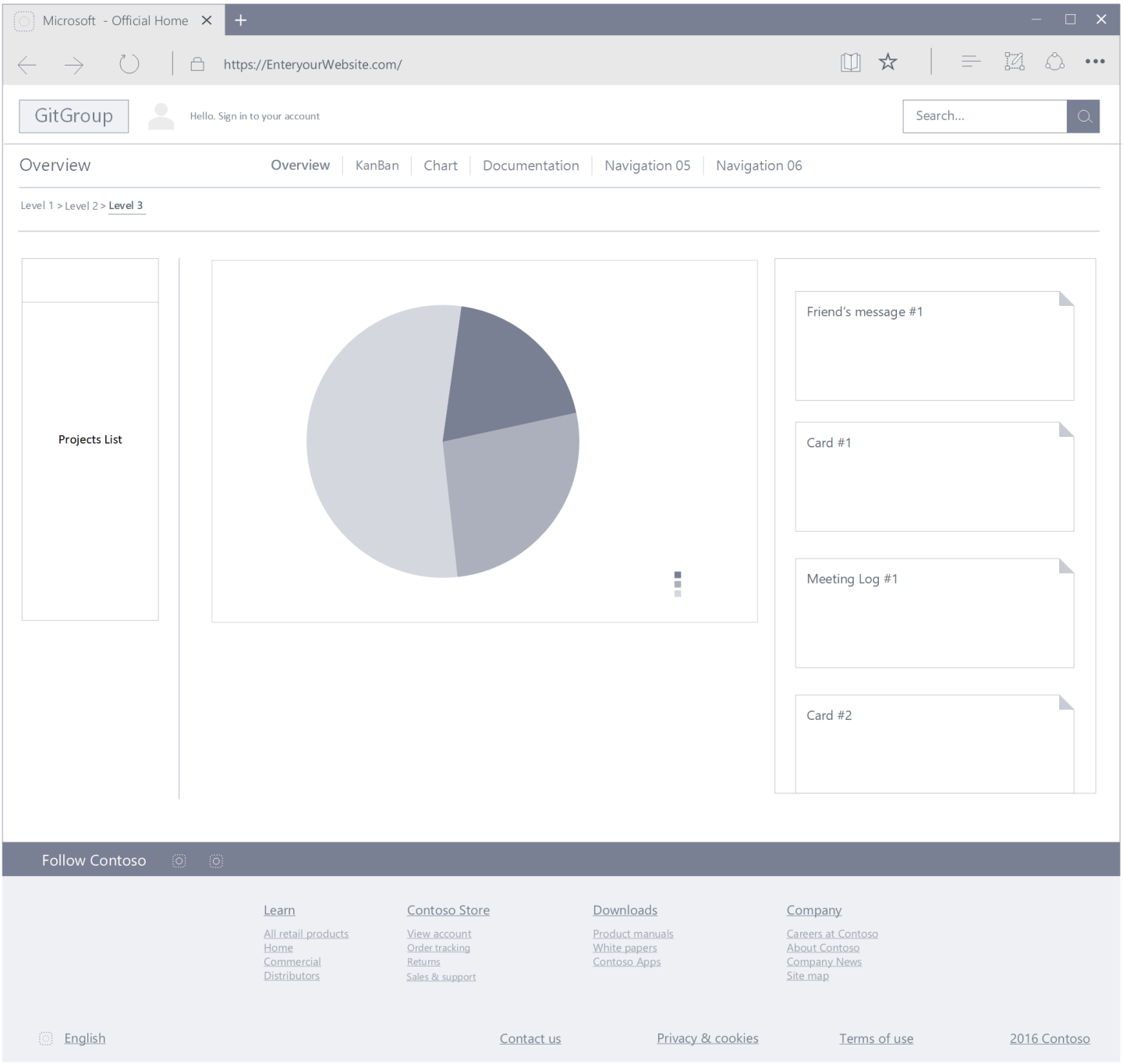

Figure A.1: **DETAIL** Overview User Interface

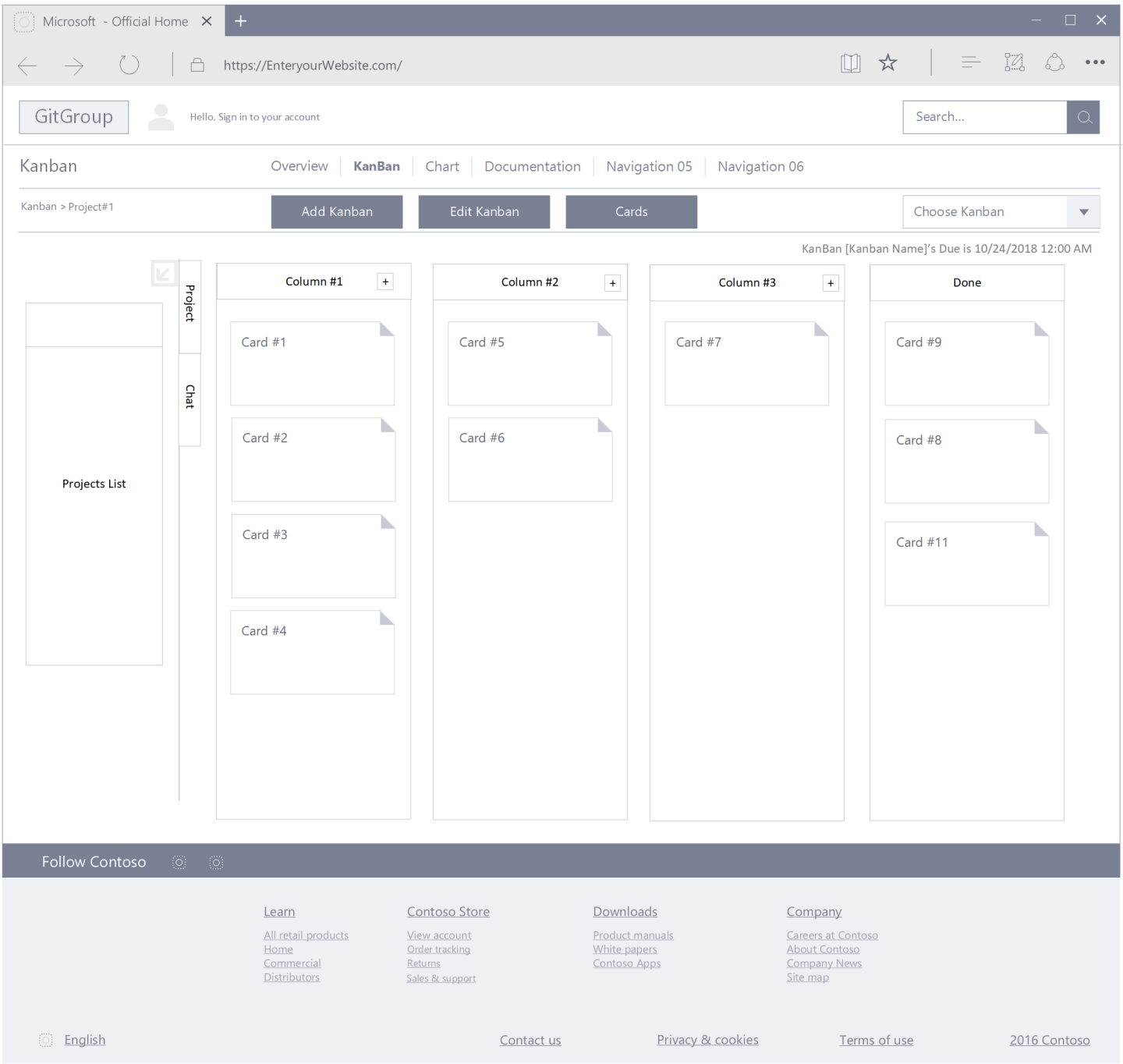

Figure A.2: **DETAIL** KanBan User Interface

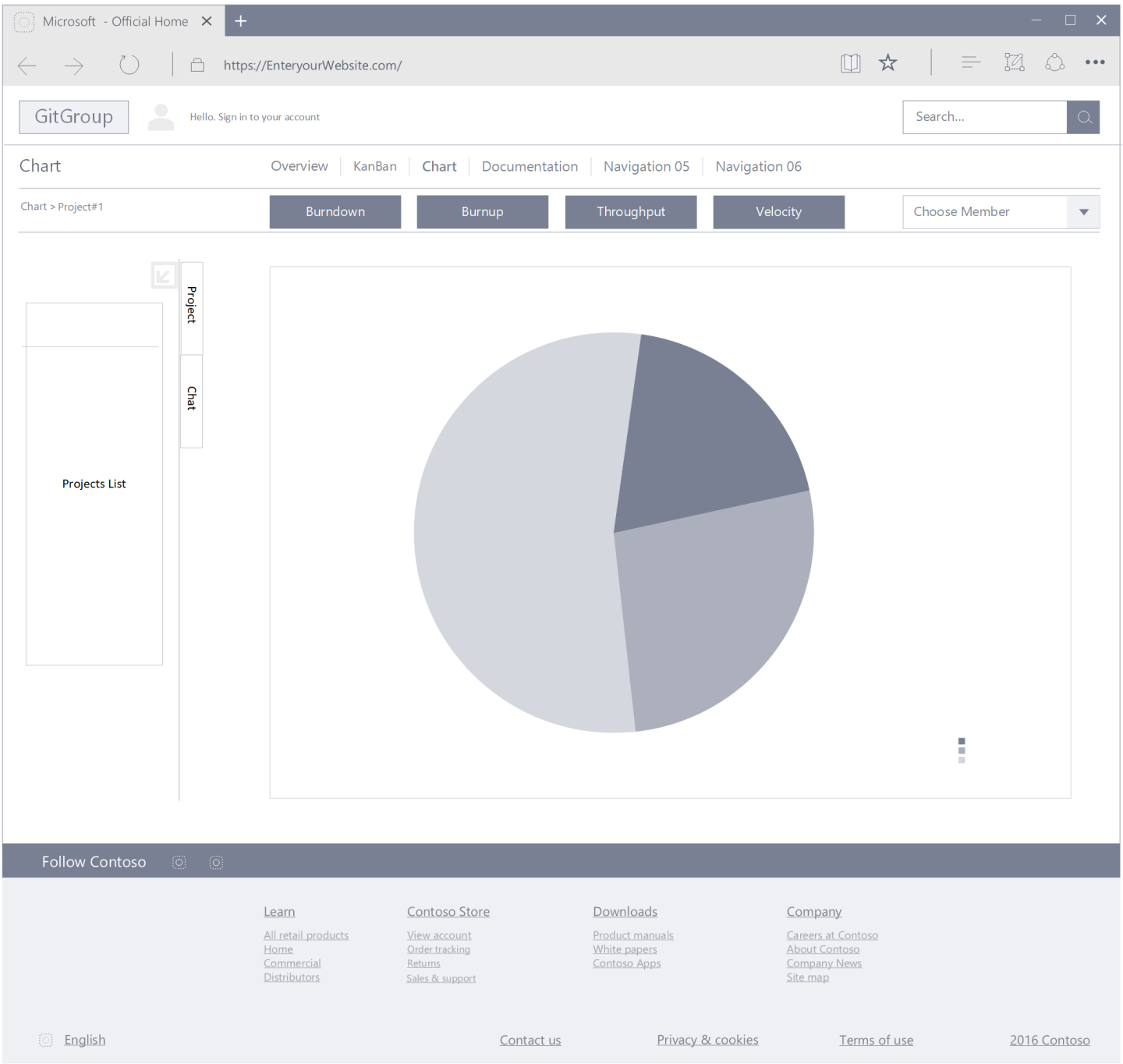

Figure A.3: **DETAIL** Chart User Interface

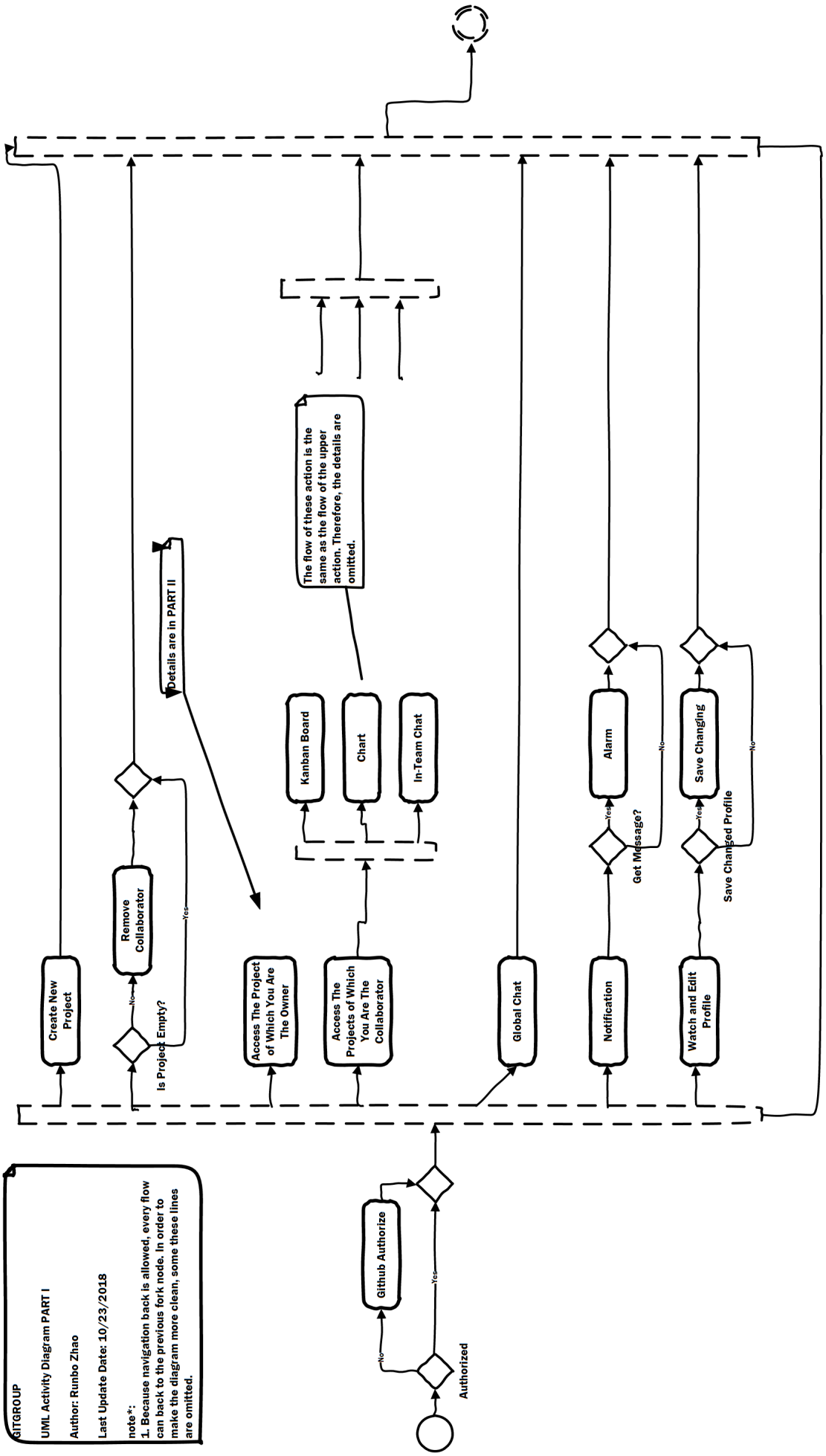

Figure A.4: **DETAIL** Activity Diagram PART I

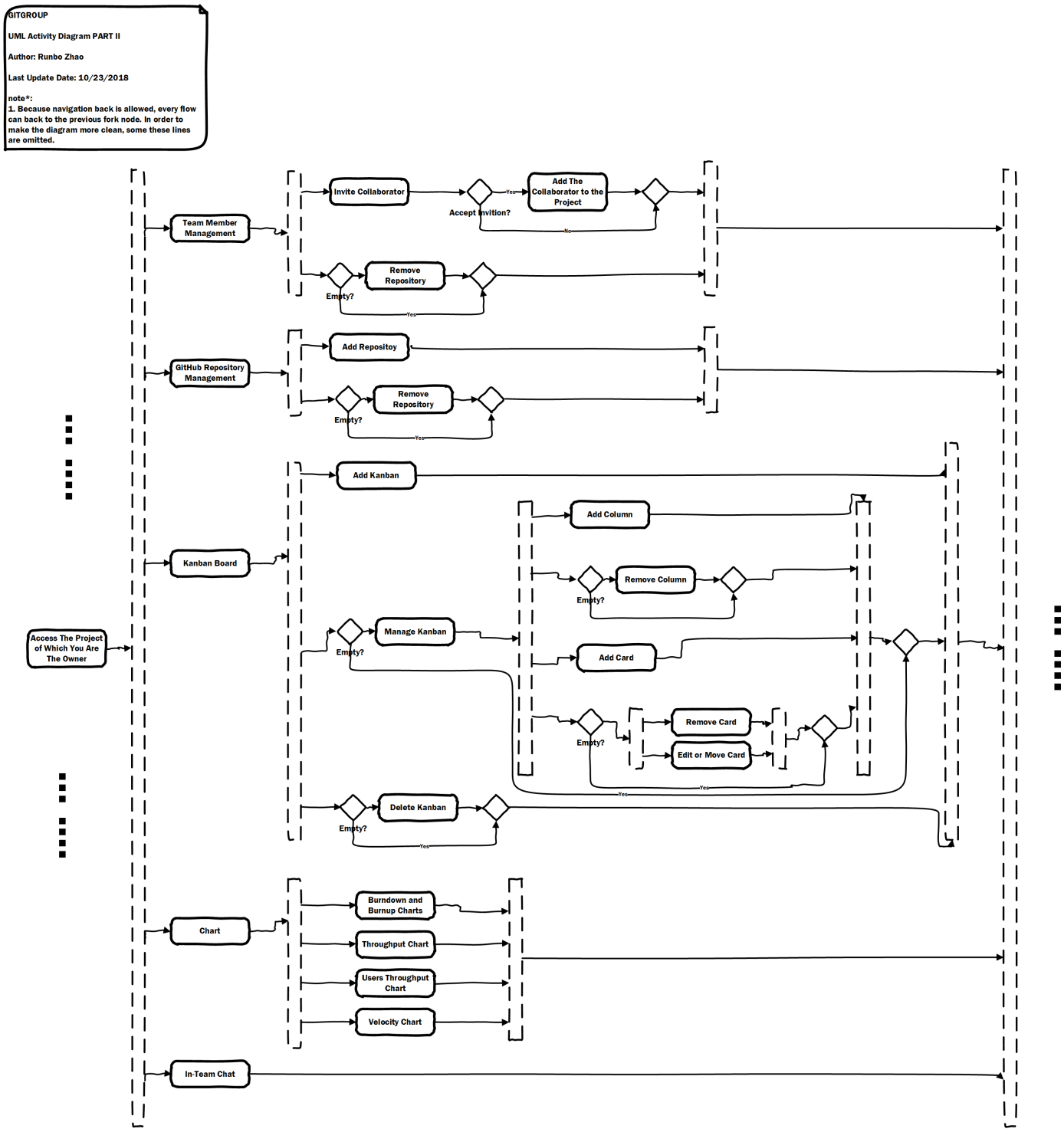

Figure A.5: **DETAIL** Activity Diagram PART II

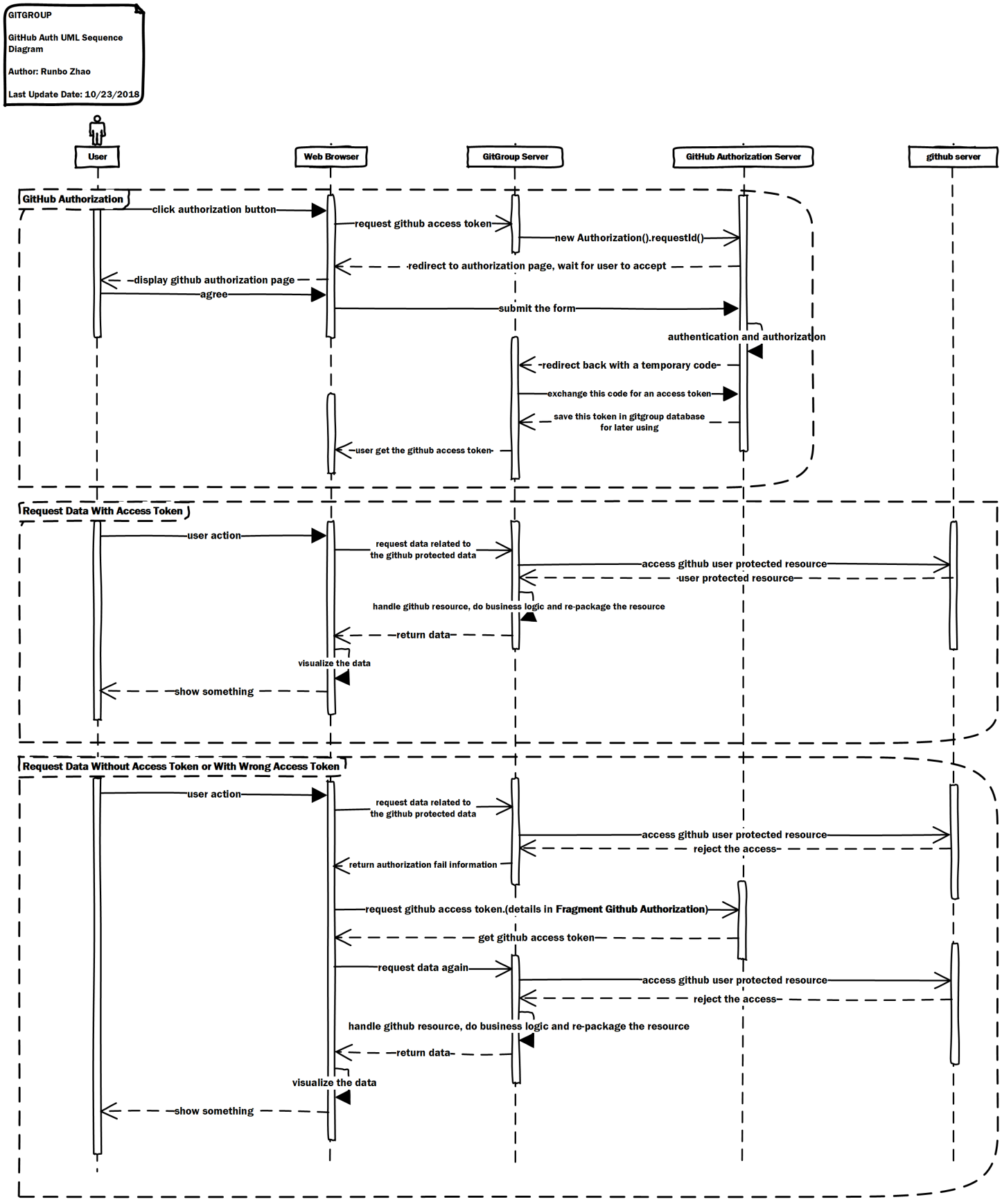

Figure A.6: **DETAIL** Sequence Diagram For Authorization

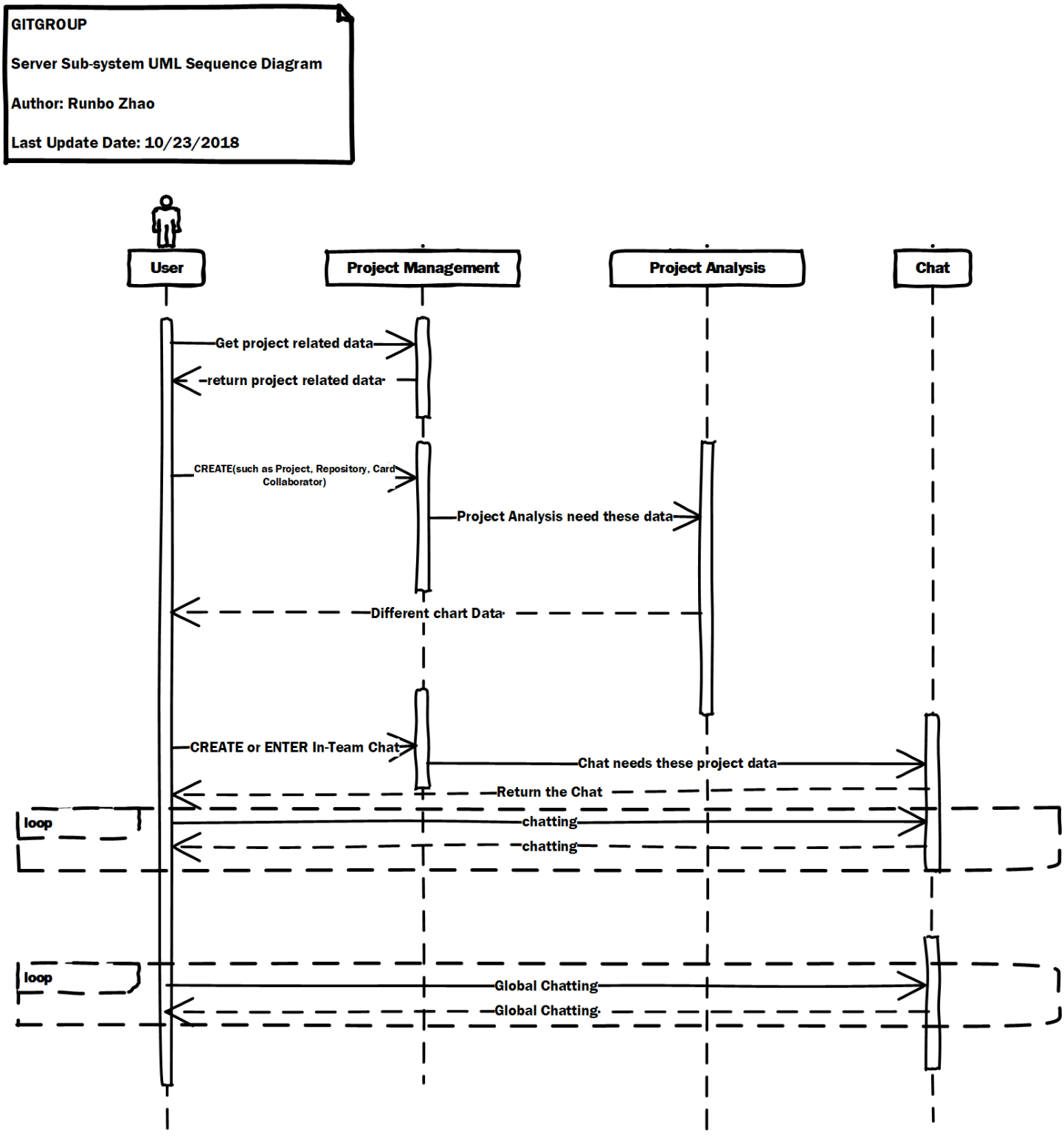

Figure A.7: **DETAIL** Sequence Diagram For Sub-Systems

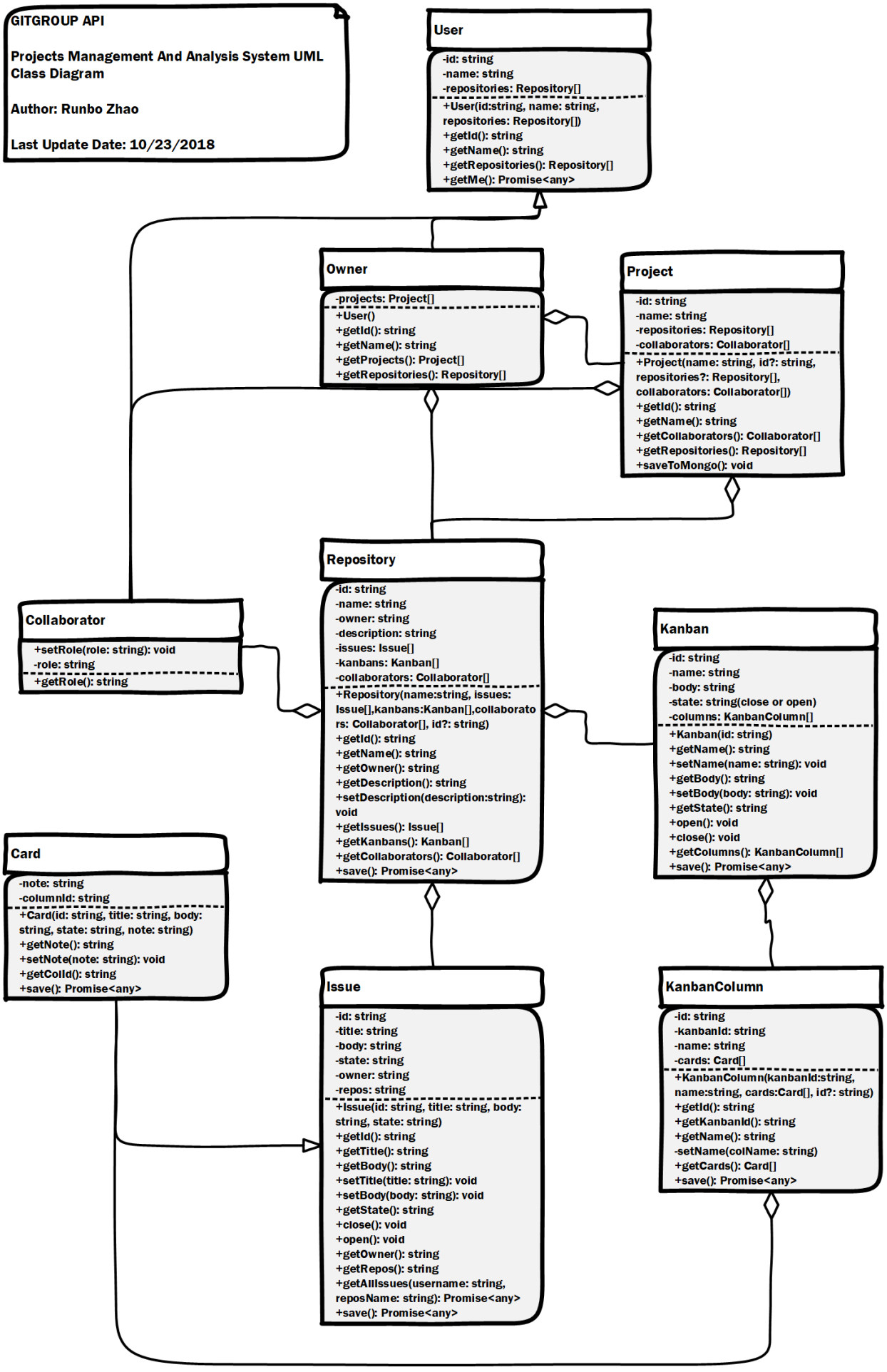

Figure A.8: **DETAIL** UML Class Diagram

List of Figures

List of Tables

1. o
2. two
3. three
4. four