SWE-573 Purposeful Community Platform

Software Requirements Specification

Yunus Sina GÜLŞEN

2017719177

### Contents:

1. Introduction  
   1.1. Purpose  
   1.2. Overview  
   1.3. Environmental Characteristics  
   1.3.1. Hardware  
   1.3.2. Software  
   1.3.3. Usability
2. Goals of Implementation
3. Overall Description  
   3.1. Product Functions  
   3.2. Platform Users and Characteristics  
   3.3. Product Glossary
4. Requirements  
   4.1. Functional Requirements  
   4.2. Non-Functional Requirements
5. Assumptions
6. Use Cases  
   6.1 Actors  
   6.2 Use Case Definitions  
   6.3 Use Case Diagrams
7. Class Diagrams
8. Sequence Diagrams  
   8.1 Sequence Diagram for 1.2.4 Create Community  
   8.2 Sequence Diagram for 1.2.9 Create Custom DataType  
   8.3 Sequence Diagram for 1.2.12 Create Post
9. Mockups  
   9.1 Register New User  
   9.2 Login  
   9.3 Browse Community  
   9.4 View Generic Post  
   9.5 View Custom DataType Post  
   9.6 New Post - Select Community  
   9.7 New Custom DataType Post  
   9.8 New Custom DataType  
   9.9 New Custom DataType Field
10. Abbreviations

## 1. Introduction

### 1.1. Purpose

This document provides an overview of the Software Requirements Specification for "SWE573 Class Project - Purposeful Community Platform" with system characteristics, description, functional and non-functional requirements, behavioral description and questions. The document aims to gather and analyze the specifications of Purposeful Community Platform.

### 1.2. Overview

The document examines the implementation goals in Section 2, overall description in Section 3, functional and non-functional requirements in Section 4.

### 1.3. Environmental Characteristics

#### 1.3.1. Hardware

There are no requirements for a bare-metal server. For ease of use, cloud-based service providers will be used. (PaaS and IaS services can be used, leaving out SaaS components)

Candidates for cloud providers:

* Microsoft azure
* Amazon Web Services

#### 1.3.2. Software

Web Server: Any container supporting Java/Python  
Programming Language: Java/Python  
Database: PostgreSQL/MongoDB  
Frontend: Angular/React/Bootstrap

In client side, the web application should be reachable from common web browsers with HTML 5.0 support.

#### 1.3.3. Usability

User interfaces of the platform should be friendly enough to enable non-technical people to use the platform from targeted client environment which can be mobile or web application.

## 2. Goals of Implementation

* Develop a timeline to insure milestone dates are met and the project is completed within the established timeline.
* Manage implementation within pre-determined time.
* Follow SDLC- Software/System Development Life Cycle
* Follow Agile project management practices.

The application development phases include requirements analysis (gathering requirements), design (high level design and detailed design), implementation (development to transfer a design to product), testing (unit and integration testing) and evolution (deployment and maintenance) phases.

## 3. Overall Description

### 3.1. Product Functions

#### 3.1.1 General

* Register new user
* Login
* Browse anonymously

#### 3.1.2 Community

* Create new community
* Delete community
* Search for a community
* View recommended communities
* Join community
* Leave community
* Create Custom DataType
* Modify own Custom DataType
* Delete own Custom DataType

#### 3.1.3 Post

* Read all posts inside a community
* Modify own posts inside a community
* Submit own posts inside a community
* Delete own posts inside a community
* Query all posts inside a community

#### 3.1.4 Administration

* Delete users
* Modify users

#### 3.1.5 Generic DataType

PCP provides a primitive datatype for all communities which is called "Generic DataType". Common fields;

* Title - used for short name, required
* Description - used for detailed description
* Semantic Tag - used for semantic search and relations, required

#### 3.1.6 Custom DataType

Community specific datatypes which inherit fields "Title" and "Semantic Tag" from primitive structure "Generic DataType". A community can have multiple DataType's. Custom DataType posts can use the following field types;

* Image
* Audio
* Video
* Text
* Enumeration
* Email
* Url
* Datetime
* Number

### 3.2. Platform Roles and Functions

#### 3.2.1 Roles

* Registered User
* Anonymous
* Administrator

#### 3.2.2 Roles & Functions

| **Function** | **Registered User** | **Anonymous** | **Administrator** |
| --- | --- | --- | --- |
| Register |  | x |  |
| Login | x |  | x |
| Browse | x | x | x |
| Create Community | x |  | x |
| Search Community | x | x | x |
| Join Community | x |  | x |
| Leave Community | x |  | x |
| Delete Community |  |  | x |
| Browse Community | x | x | x |
| View Recommended Community | x | x | x |
| Create Custom DataType | x |  | x |
| Modify own Custom DataType | x |  | x |
| Delete own Custom DataType | x |  | x |
| Read Posts | x | x | x |
| Submit Post | x |  | x |
| Modify Own Posts | x |  | x |
| Delete Own Posts | x |  | x |
| Query All Posts | x | x | x |
| Delete Users |  |  | x |
| Modify Users |  |  | x |

### 3.3. Product Glossary

* Community - A group of DataType's with subscribed users having a particular characteristic in common
* DataType - A structure to hold information with predefined fields
* Community Builder - Community User's who create new communities or custom data types
* Community User - Registered user who has "Registered User" role
* Anonymous User - Unregistered users who has "Anonymous" role
* Field - Representation of data in specific types
* Primitive Data Type - A basic structure to hold the predefined format of a community (Title, Description, Semantic Tag)
* Generic DataType - Initial common datatype and official channel of a community
* Custom DataType - A predefined structure and sub category to gather data from community users in a structured way

## 4. Requirements

### 4.1. Functional Requirements

| **Requirement** | **Short Description** | **Perspective** | **Details** |
| --- | --- | --- | --- |
| 4.1.1 | Register | User | Users shall be able to register to the platform if they need to join communities and create custom DataType's. Users shall be able to register by providing First name, Last name, username, password details. If all fields are valid, then the user should be saved to the user table in database. After successful registration, platform should redirect to login page. |
| 4.1.2 | Database | Product | Database used for the platform shall be MongoDB or PostgreSQL |
| 4.1.3 | Programming Language | Product | Programming language used to implement the platform shall be Java or Python |
| 4.1.4 | Roles | User | Platform shall provide users two user roles. 1) Community builder - Can define data types, create new communities 2) Community user - Can join communities, can create content on generic or predefined custom data types |
| 4.1.5 | Content Evolution | User | Platform shall provide a way to evolve stored information. User created custom DataType's shall be evolved by adding or removing fields |
| 4.1.6 | Generic Data Type | Product | Generic DataType shall have Title, Description and Semantic Tag fields |
| 4.1.7 | Custom Data Types | Product | Custom DataType's shall inherit Title and Semantic Tag fields. Community builders shall be able to customize custom DataType's from predefined primitive types |
| 4.1.8 | Recommendation | User | Platform shall provide users a way to relate communities or posts with recommendations from wikidata services. Platform shall provide suggestions to relate information to WikiData content using entered query text |
| 4.1.9 | Front-end | Product | Platform front-end shall be mobile or web application |
| 4.1.10 | Full-Text Search | User | Platform shall provide a full-text search function for users to search stored generic information inside communities. Users shall be able to query generic content with full-text search |
| 4.1.11 | Filtered Search | User | Platform shall provide a custom search function for users to search stored information inside communities. Users shall be able to query stored custom content with filters on fields |
| 4.1.12 | Data Types | User | Platform shall provide users a way to create Custom DataType's using primitive types Image, Audio, Video, Text, Enumaration, Email, Url, Datetime, Number} |

### 4.2. Non-Functional Requirements

| **Requirement No** | **Short Description** | **Type** | **Details** |
| --- | --- | --- | --- |
| 4.2.1 | WikiData Tag Selection | Usability | Platform shall provide an easy to use way to select from recommended WikiData tags when entering Semantic Tag field data |

## 5. Assumptions

| **Assumption No** | **Short Description** | **Details** |
| --- | --- | --- |
| 5.1 | Alter Custom DataType | When builders alters Custom DataType definition, the platform shall behave optimistic about orphaned information |

## 6. Use Cases

### 6.1 Actors

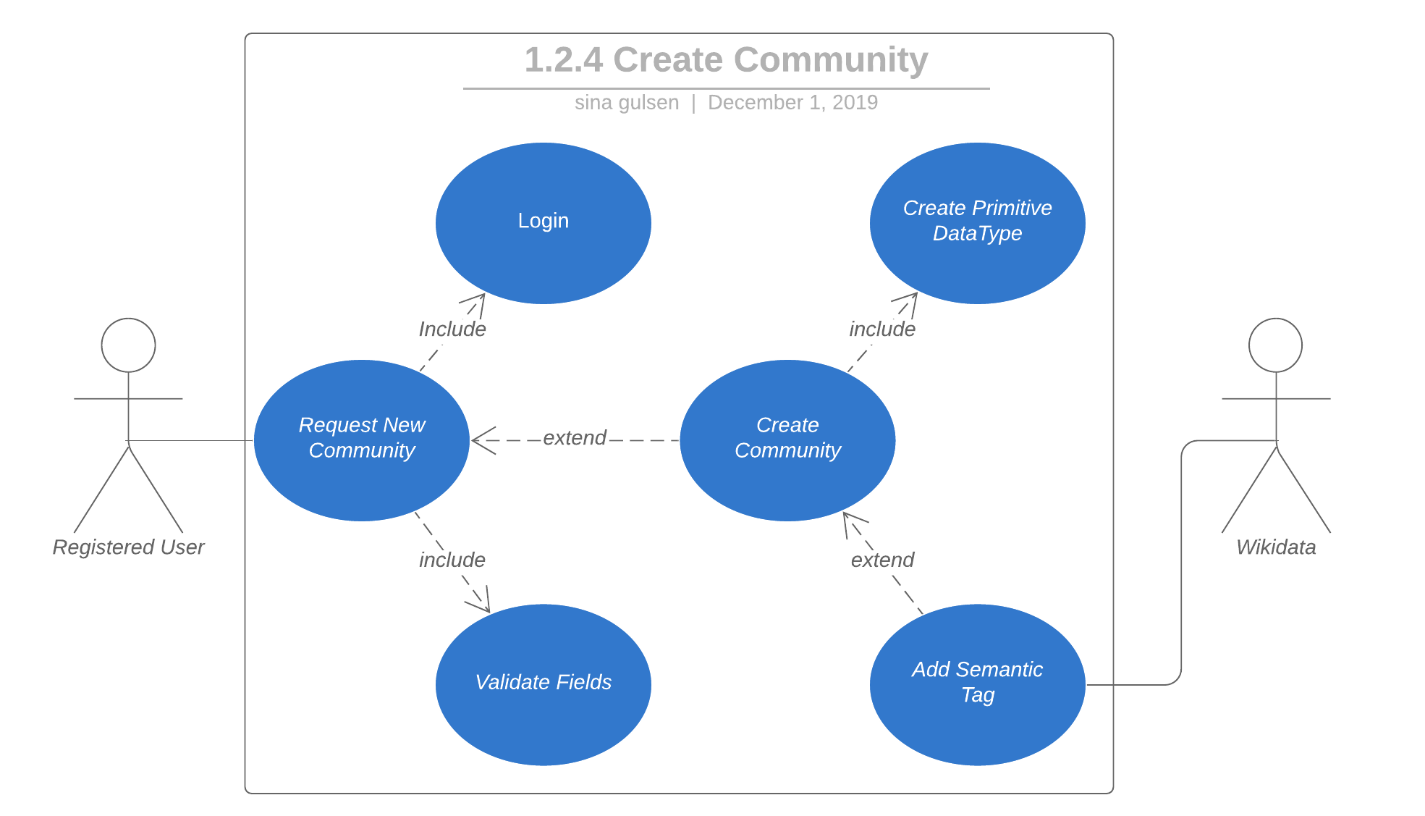
| **Id** | **Name** |
| --- | --- |
| 1.1.1 | Administrator |
| 1.1.2 | Anonymous |
| 1.1.3 | Registered User |
| 1.1.4 | Wikidata |

### 6.2 Use Case Definitions

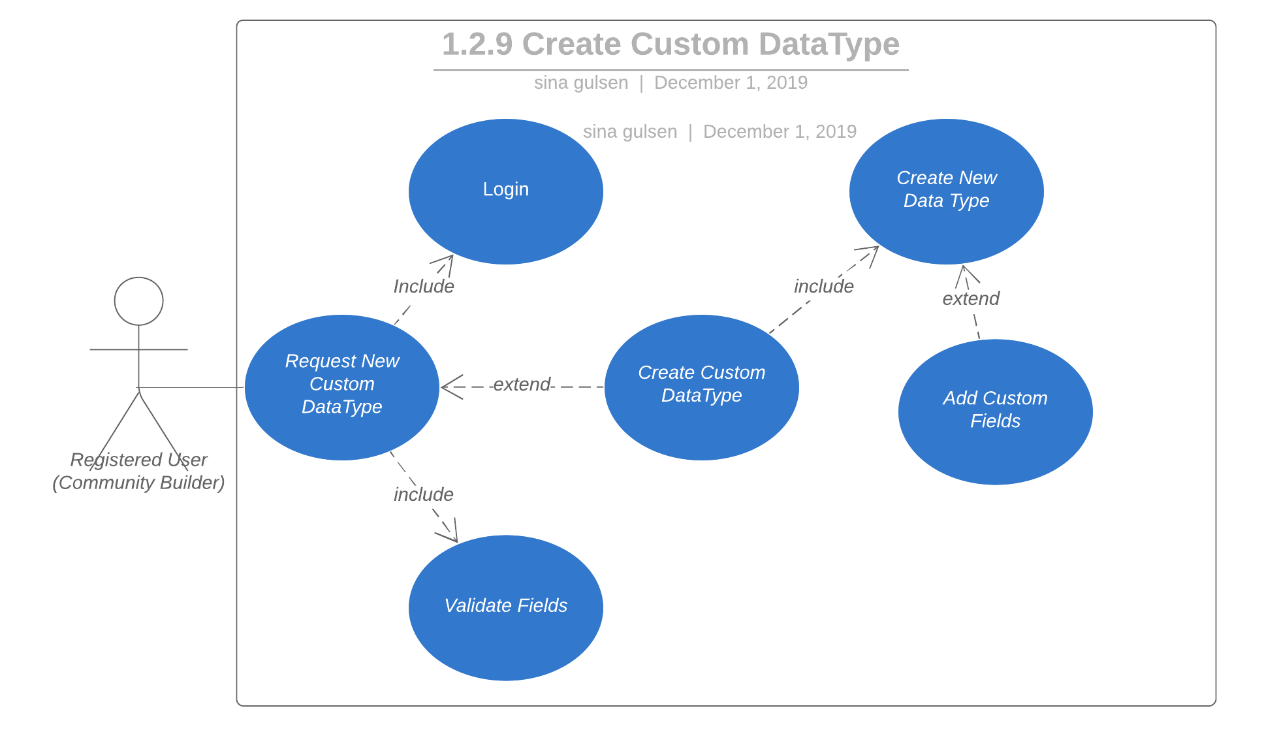
| **Id** | **Name** |
| --- | --- |
| 1.2.1 | Create User |
| 1.2.3 | Login |
| 1.2.4 | Create Community |
| 1.2.5 | Browse Community |
| 1.2.6 | Join Community |
| 1.2.7 | Leave Community |
| 1.2.9 | Create Custom DataType |
| 1.2.10 | Modify Custom DataType |
| 1.2.12 | Create Post |
| 1.2.13 | Modify Post |
| 1.2.14 | Browse Posts |
| 1.2.15 | Delete Posts |
| 1.2.16 | Delete Users |
| 1.2.17 | Modify Users |
| 1.2.18 | Get Semantic Tags |
| 1.2.19 | Get Recommended Communities |
| 1.2.20 | Generic Search |
| 1.2.21 | Custom Search |

### 6.3 Use Case Diagrams

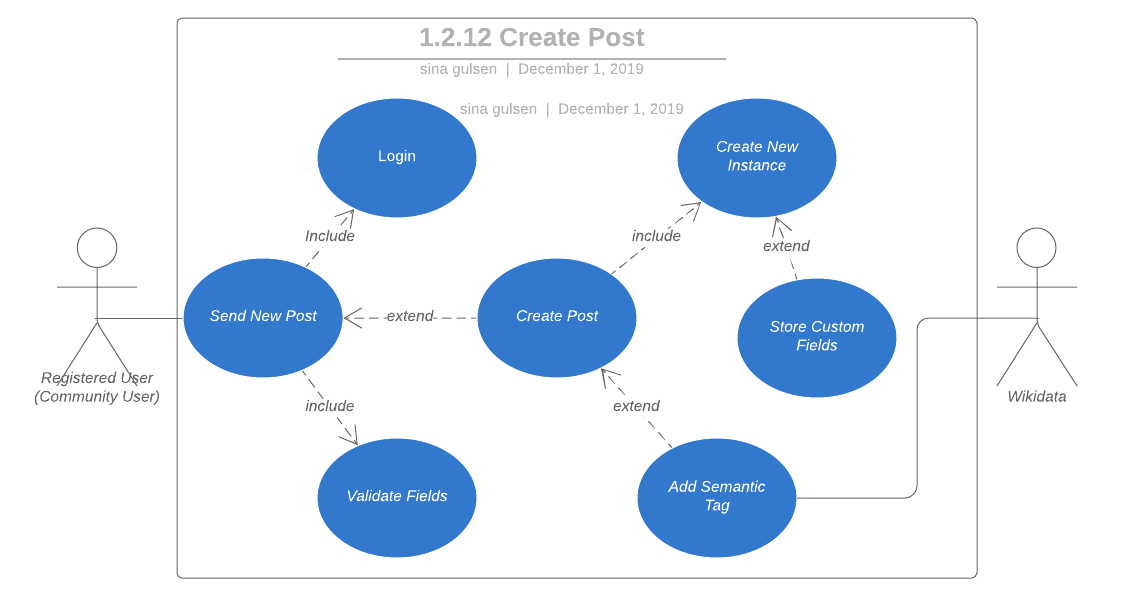
#### 6.3.1 Create Community (Use case 1.2.4)



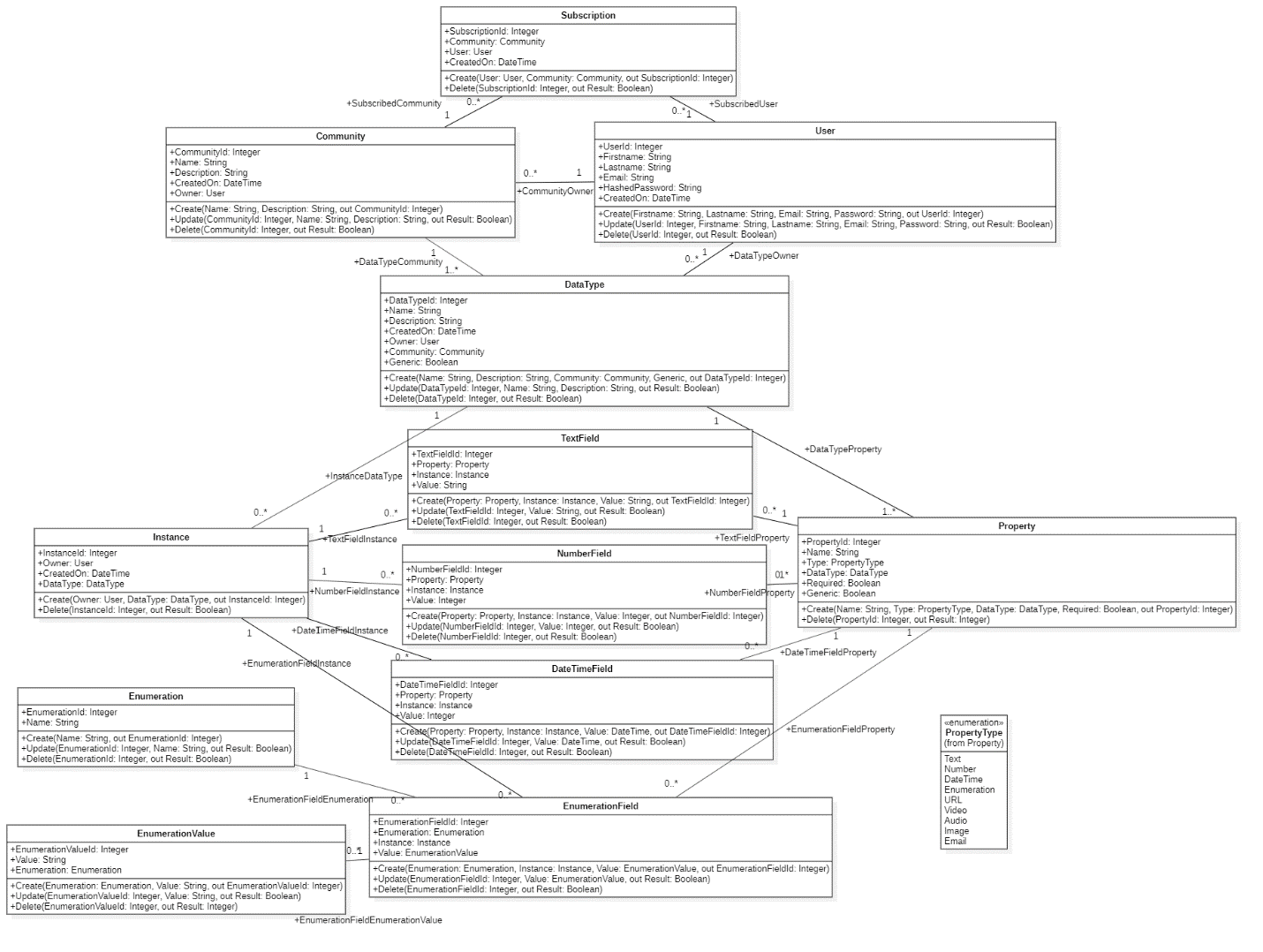
#### 6.3.2 Create Custom DataType (Use case 1.2.9)



#### 6.3.3 Create Post (Use case 1.2.12)

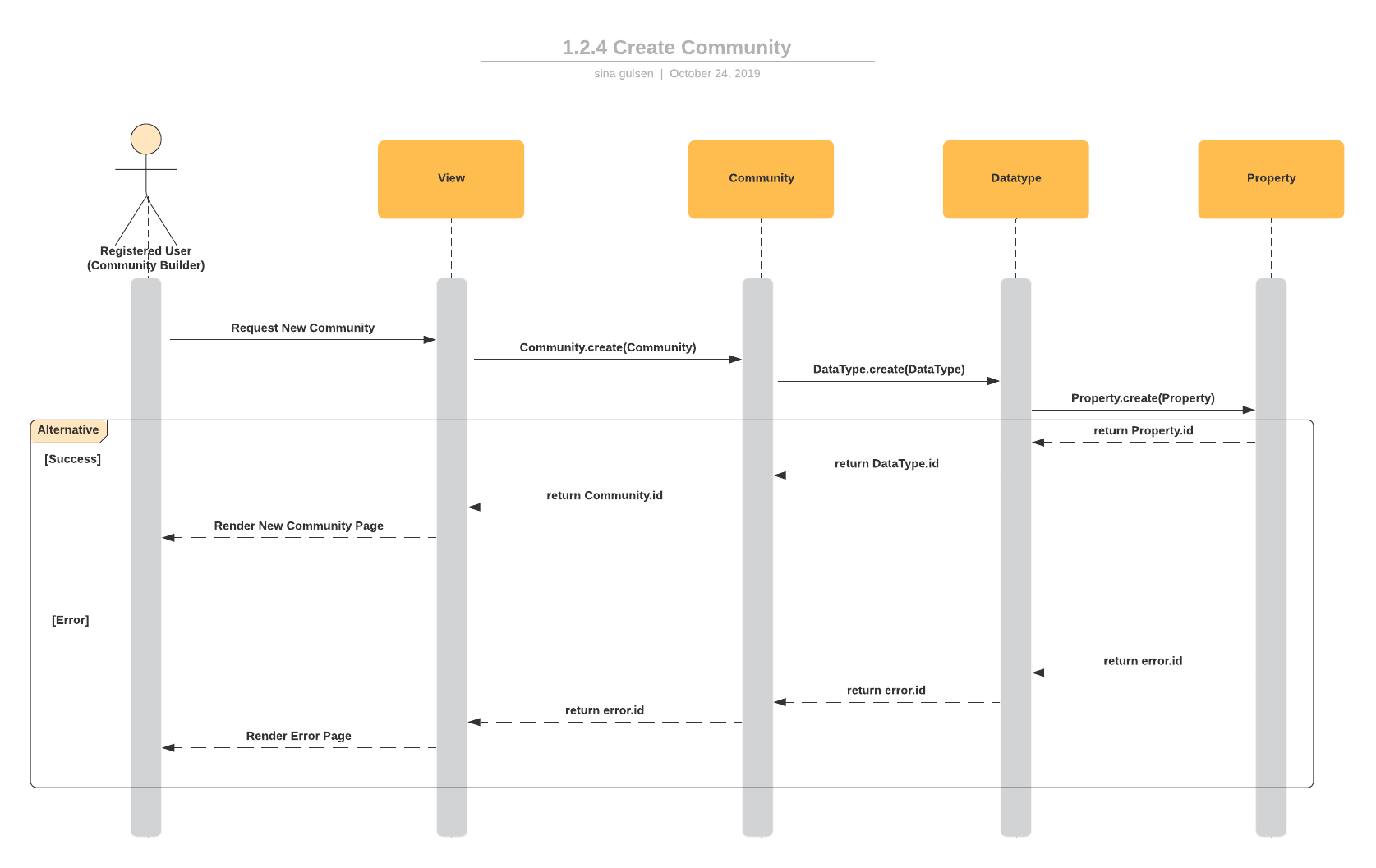


## 7. Class Diagrams

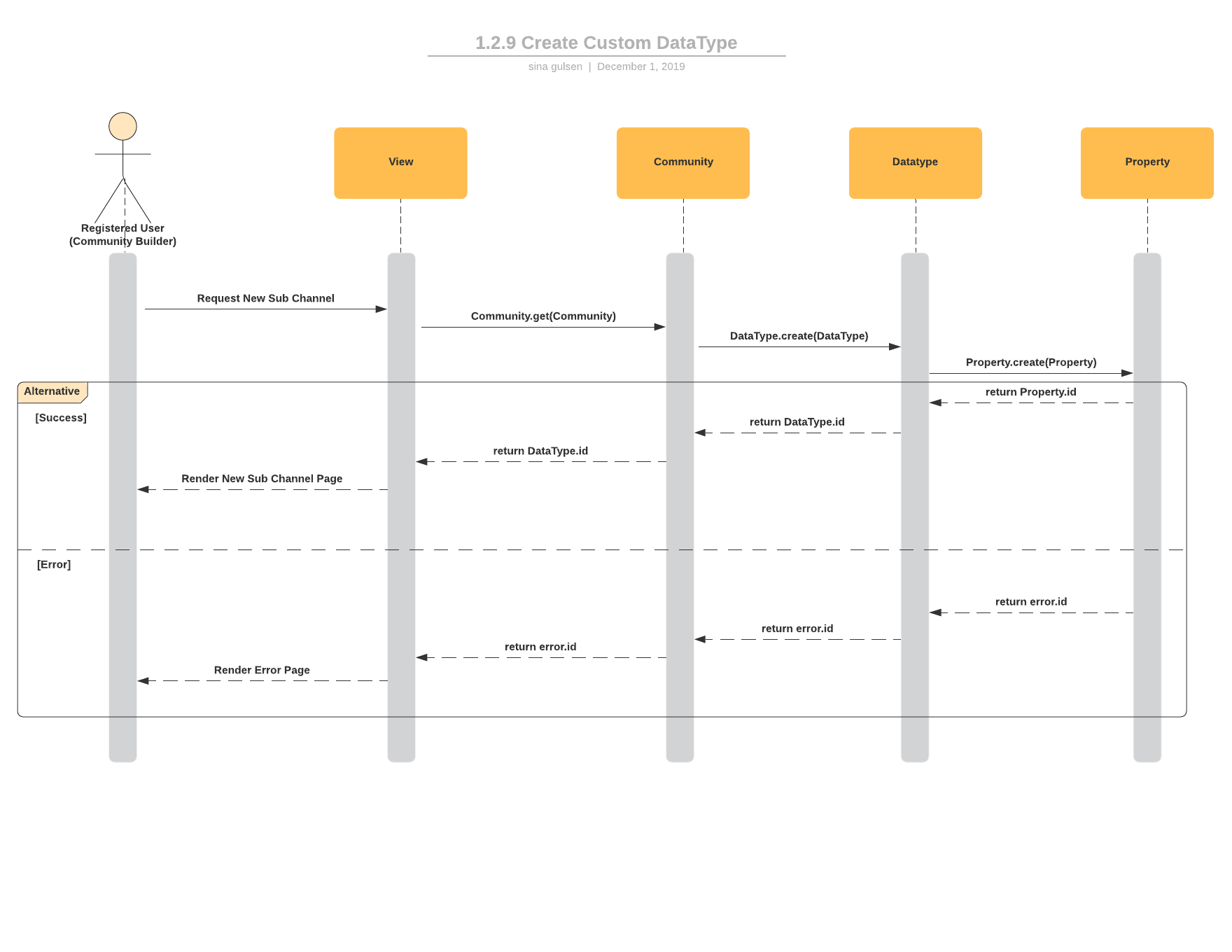
****

## 8. Sequence Diagrams

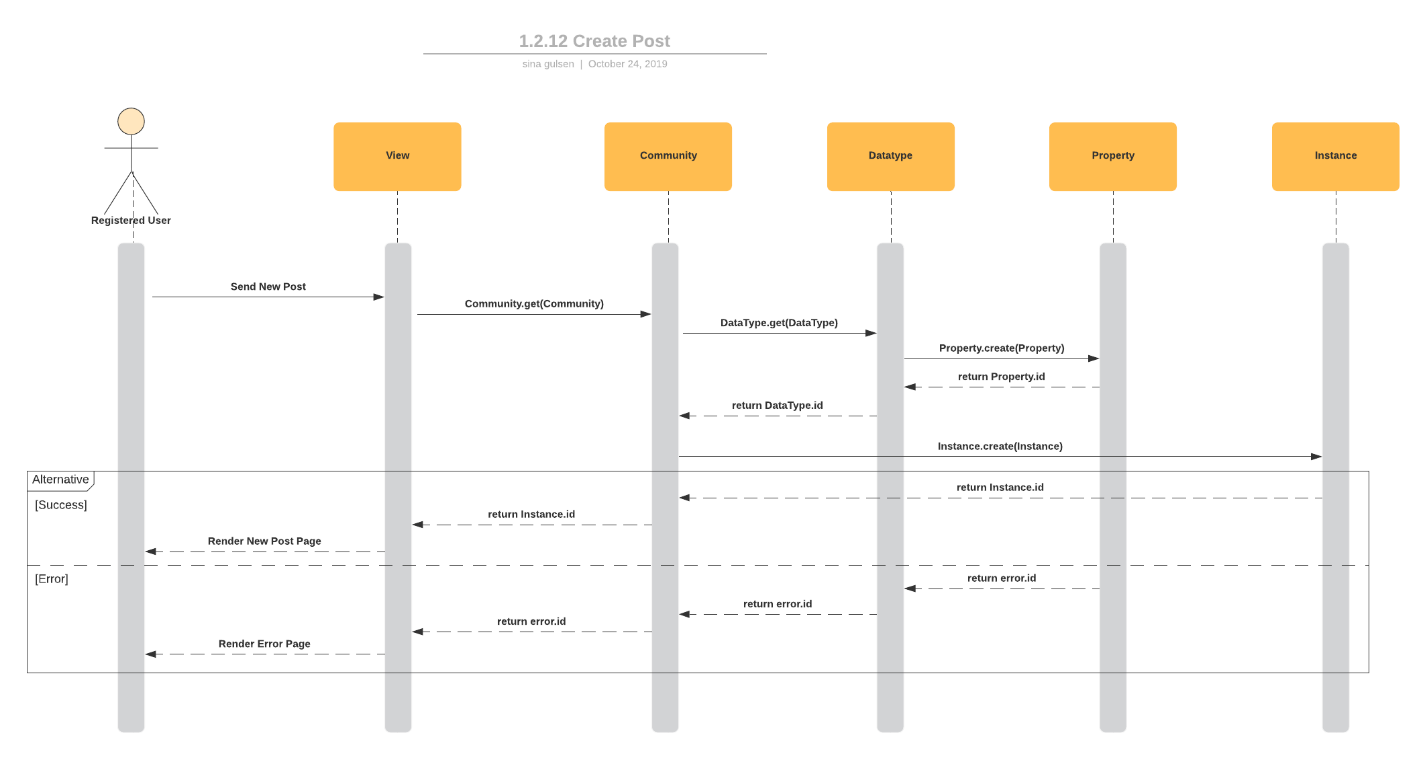
### 8.1 Sequence Diagram for 1.2.4 Create Community



### 8.2 Sequence Diagram for 1.2.9 Create Custom DataType

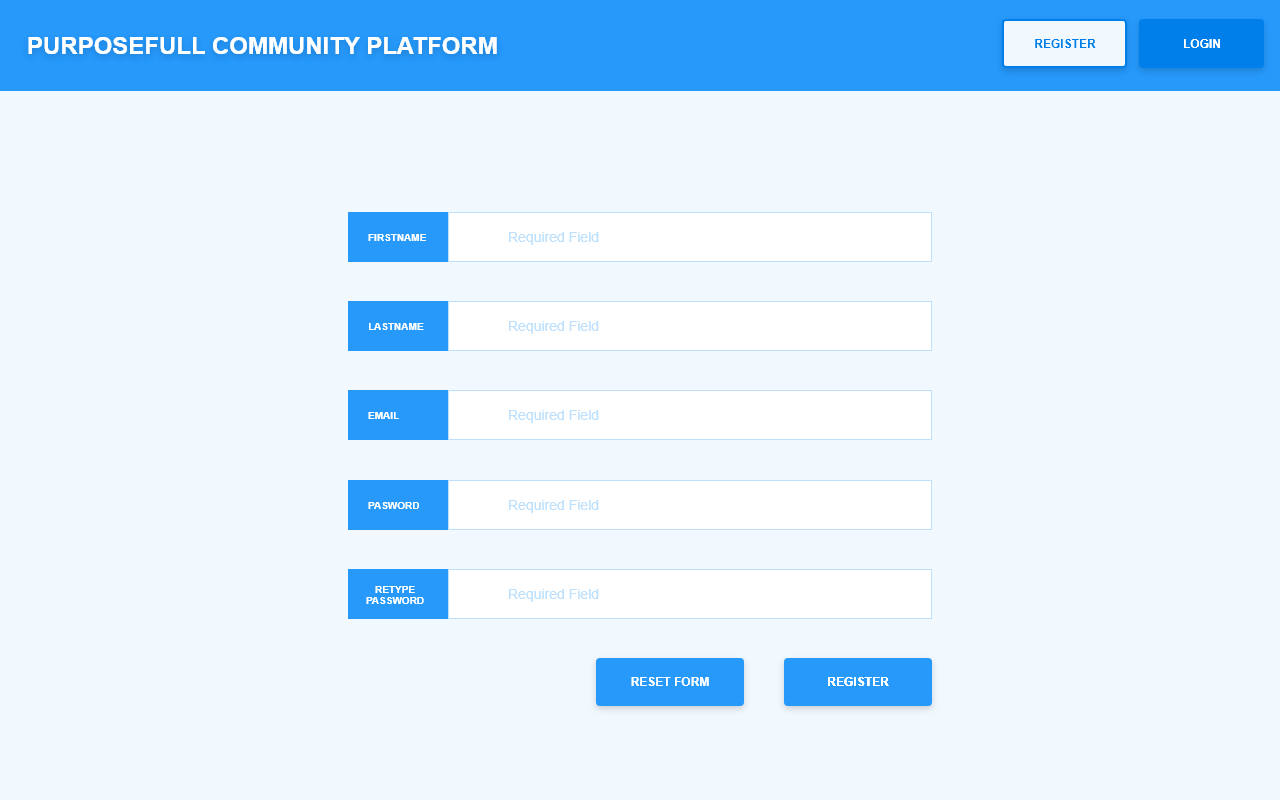


### 8.3 Sequence Diagram for 1.2.12 Create Post

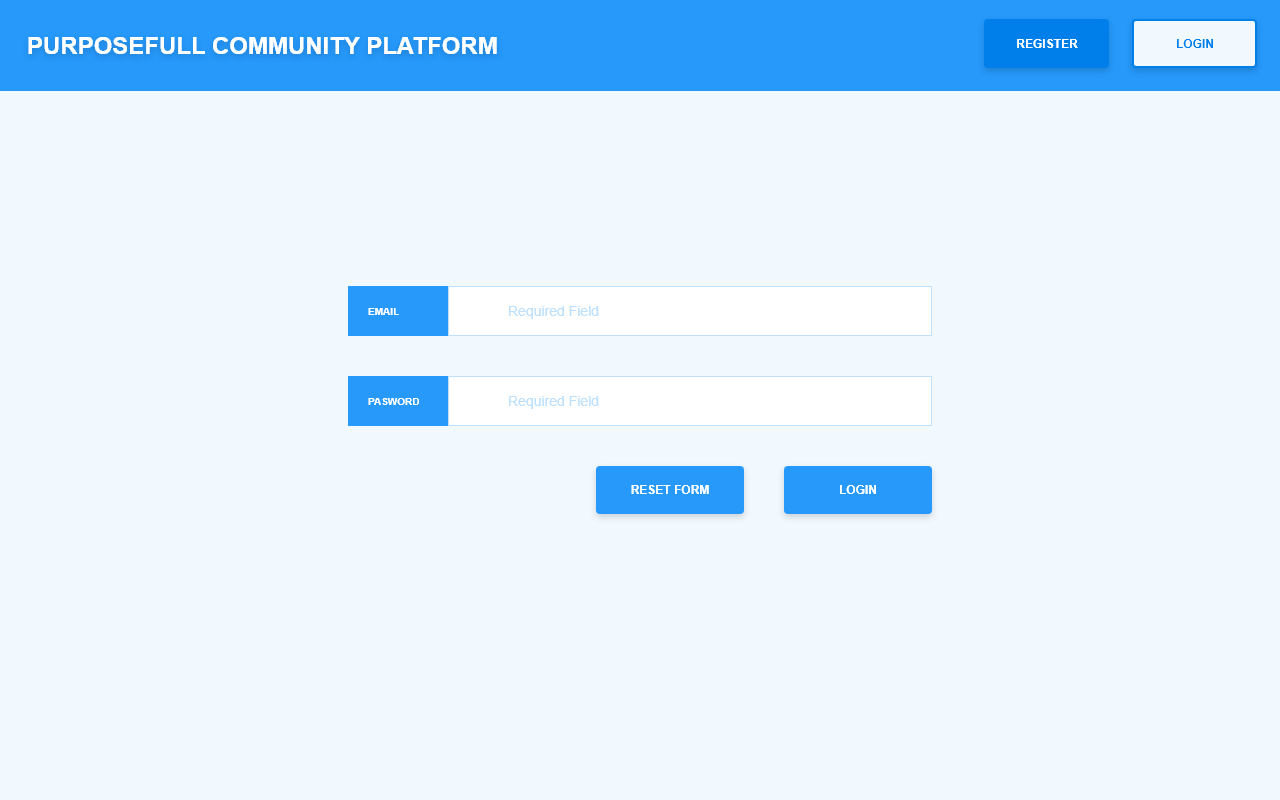


## 9. Mockups

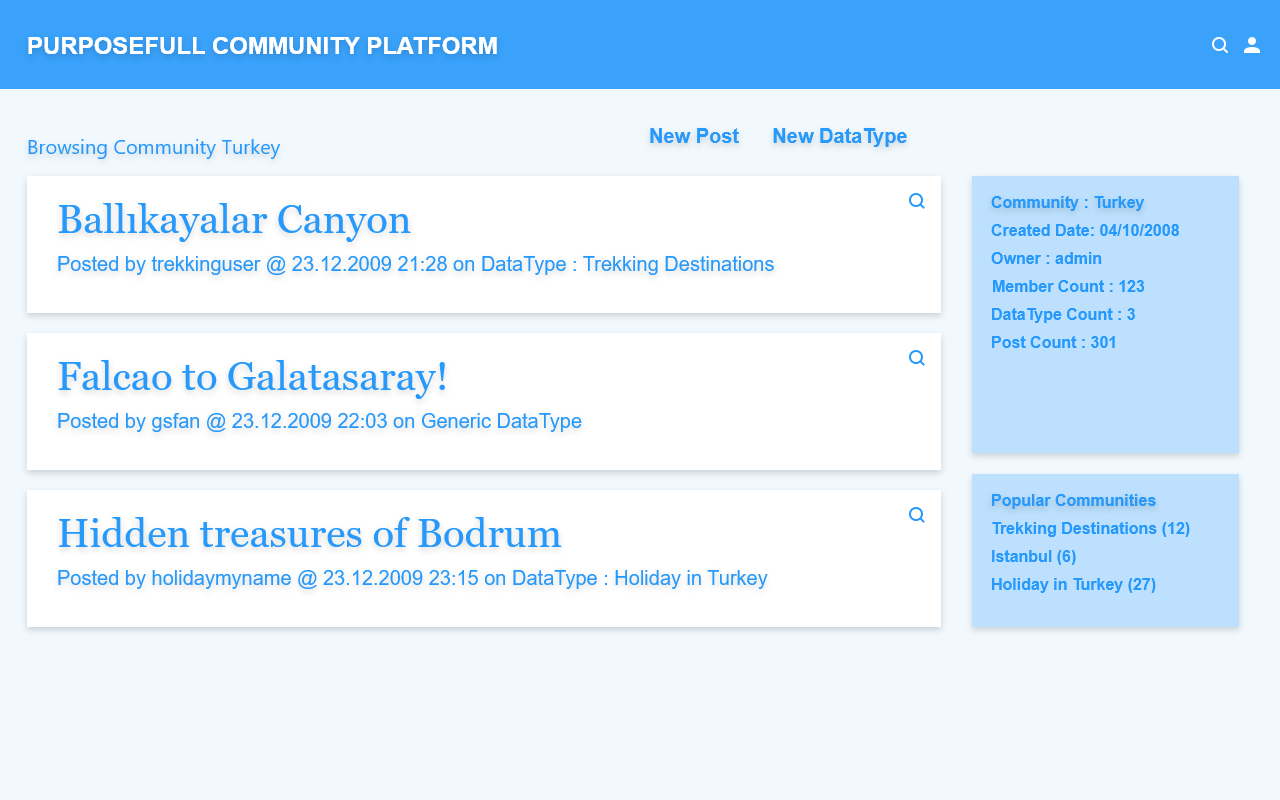
### 9.1 Register New User



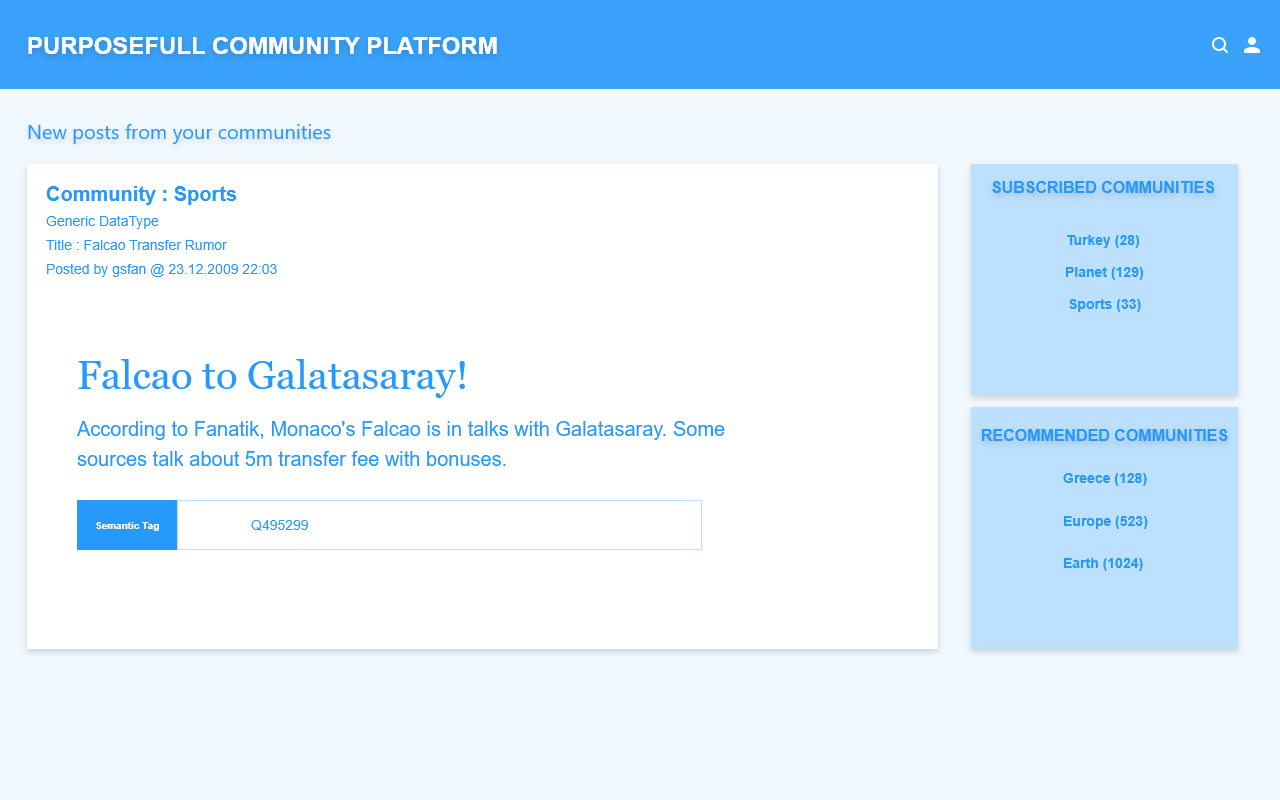
### 9.2 Login



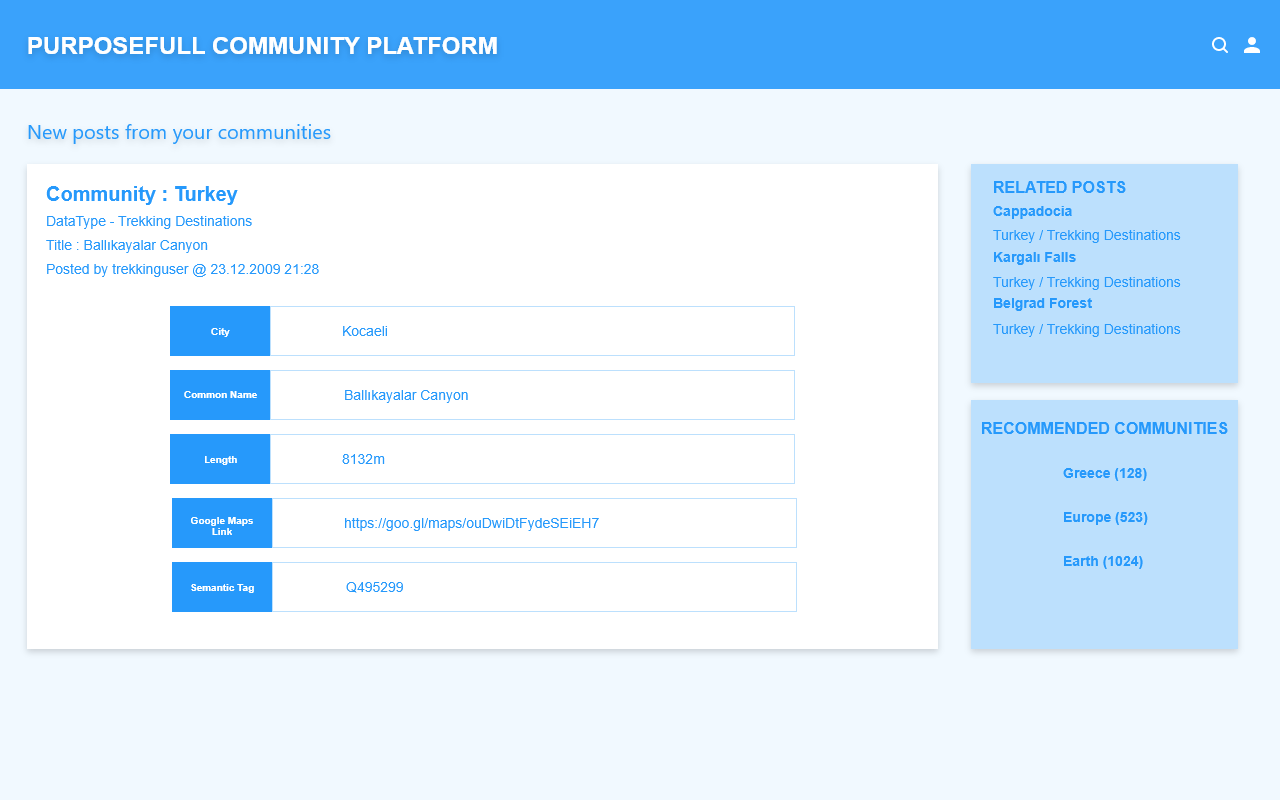
### 9.3 Browse Community



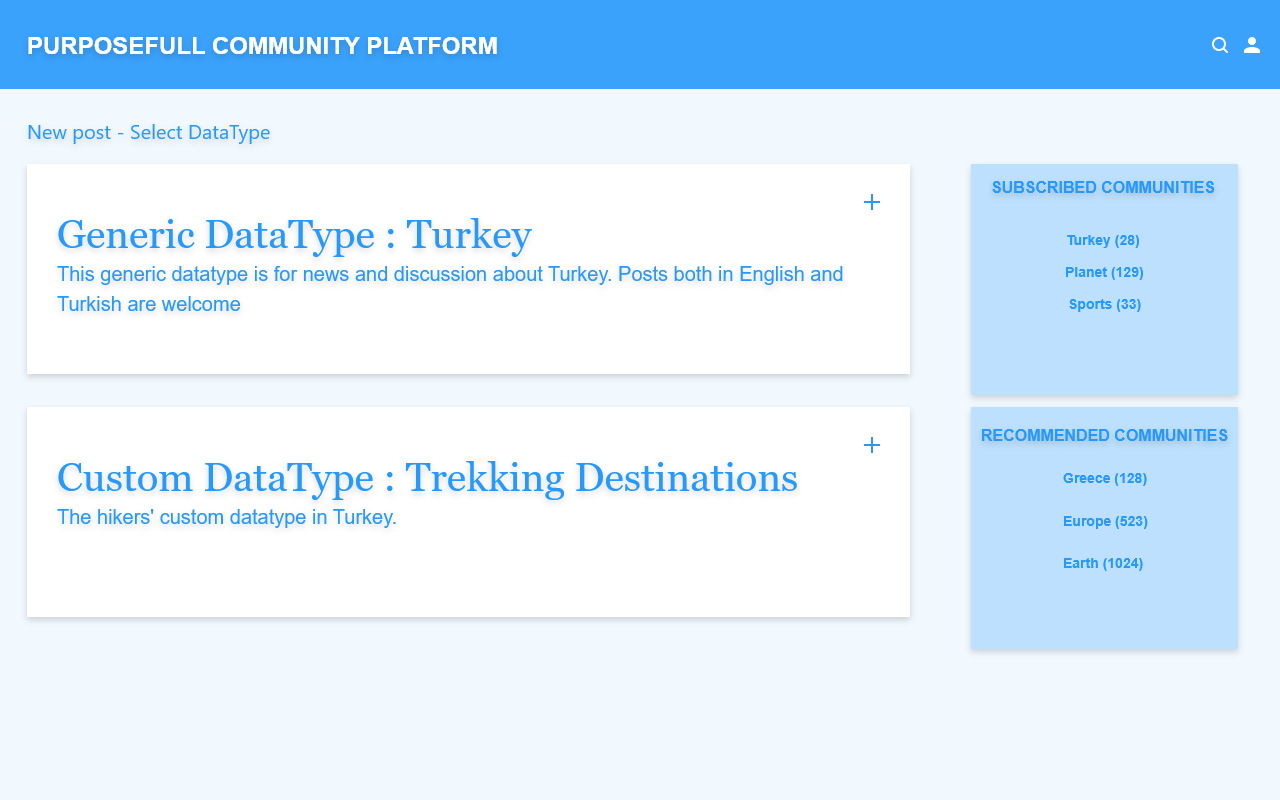
### 9.4 View Generic Post



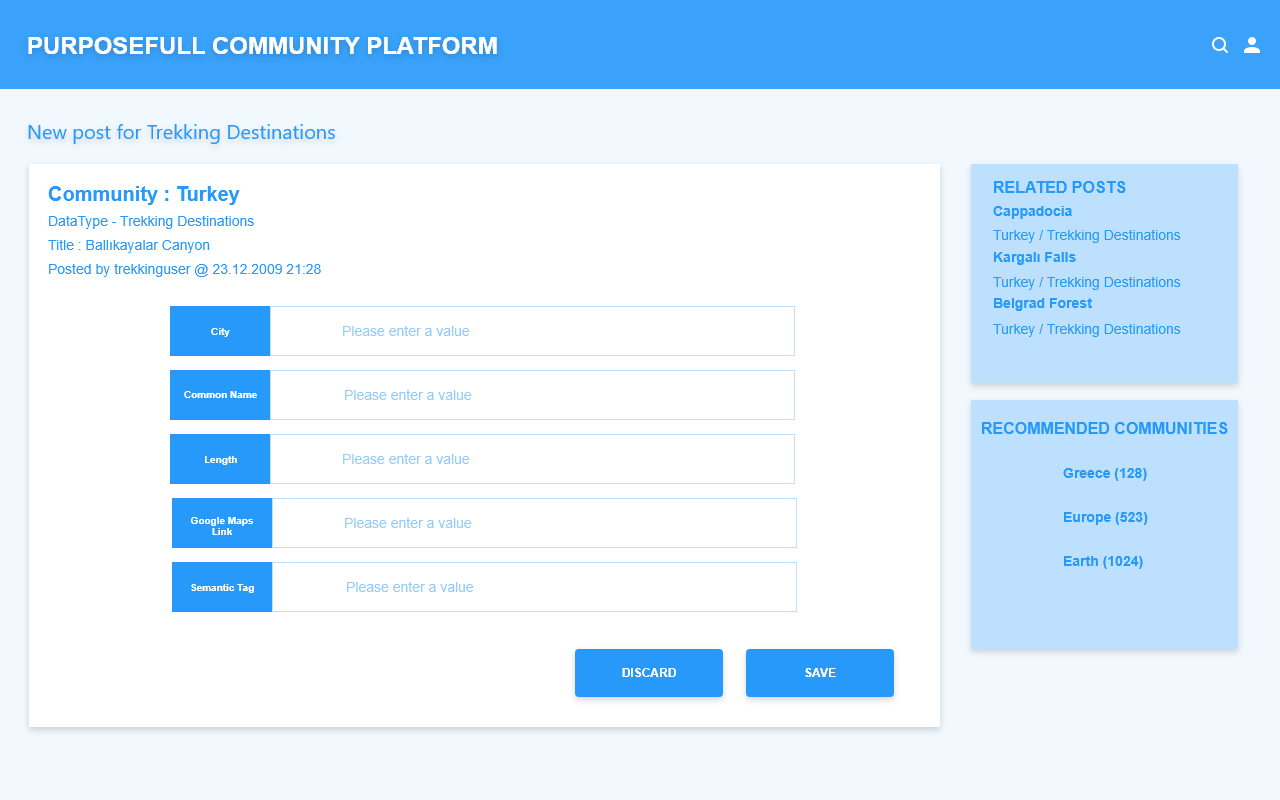
### 9.5 View Custom DataType Post



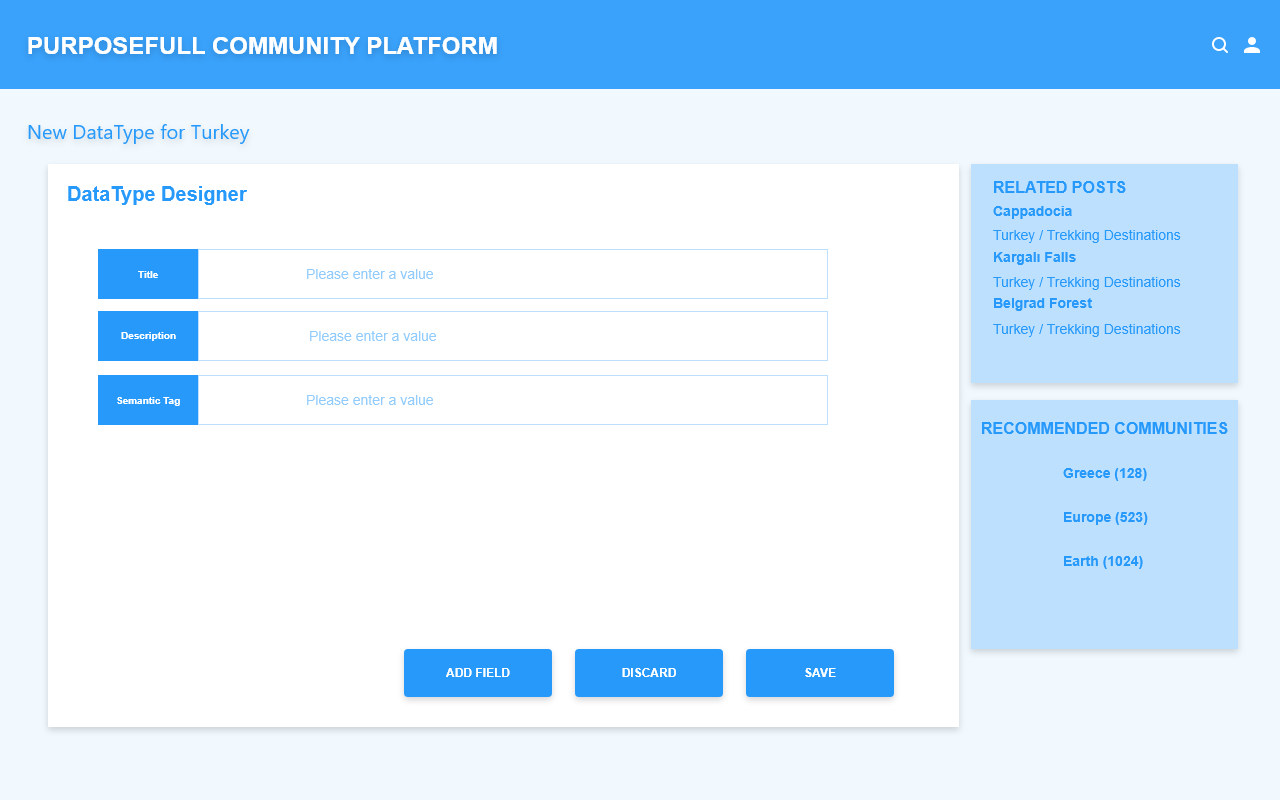
### 9.6 New Post - Select Community



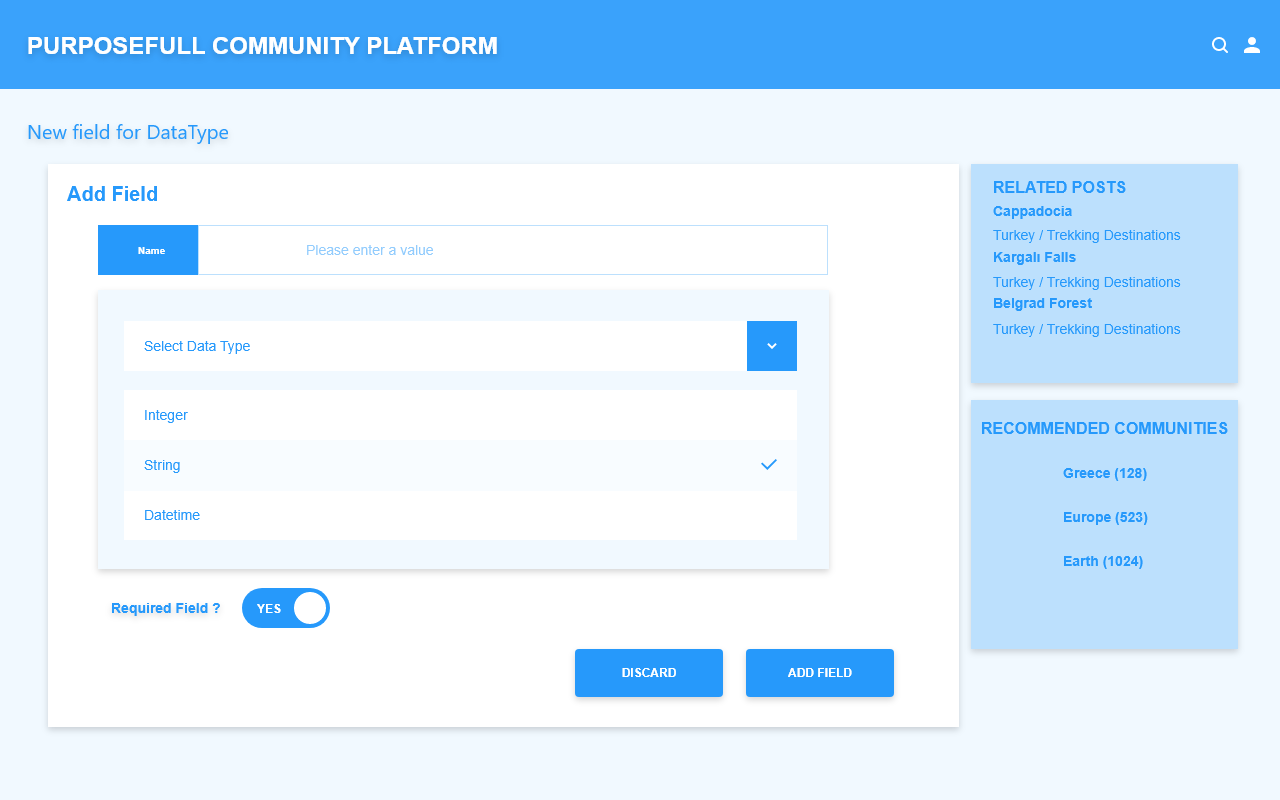
### 9.7 New Custom DataType Post



### 9.8 New Custom DataType



### 9.9 New Custom DataType Field



## 10. Abbreviations

List of commonly used abbreviations used in this documentation.

* IAS - Infrastructure as a service
* PaaS - Platform as a service
* SaaS - Software as a service
* SDLC - Software development life cycle
* PDT - Primitive data type
* CSDT - Community specific data type
* PCP - Purposeful Community Platform