# System and Software Architecture Description (SSAD)

**ShareWeb**

**Team 05**

**Xuan Wang: Project Manager, Life Cycle Planner**

**LiangHao Gao: Implementation Team member**

**Xi Chen: Implementation Team member, UML Modeler, Tester**

**Yuxuan Li: Operational Concept Engineer, Prototyper, System Architect**

**Zhangbiaoge Tian: Implementation Team, NDI/NCS Acquier, Evaluator**

**Chuhan Zheng: Feasibility Analyst, Requirements Engineer**

**Sahar Pourmohammadhosseini: IIV&V, Quality Focal Point**

**12/05/16**

# Version History

| Date | Author | Version | Changes made | Rationale |
| --- | --- | --- | --- | --- |
| 10/12/16 | Xi Chen | 1.0 | * Complete section 1 and 2 of the document. | * Initial draft for use with Instructional   ICM-Sw v1.0 |
| 11/16/16 | Xuan Wang | 1.5 | * Add Architectural styles, patterns and frameworks section, update class diagram and ER diagram. | * Complete overall Architecture |
| 12/5/16 | Yuxuan Li | 2.0 | * Complete section 3 | * Final document for delivery |

# Table of Contents

System and Software Architecture Description (SSAD) i

Version History ii

Table of Contents iii

Table of Tables iv

Table of Figures vi

1. Introduction 1

1.1 Purpose of the SSAD 1

1.2 Status of the SSAD 1

2. System Analysis 2

2.1 System Analysis Overview 2

2.2 System Analysis Rationale 16

3. Technology-Independent Model 17

4. Technology-Specific System Design 18

4.1 Design Overview 18

4.2 Design Rationale 24

5. Architectural Styles, Patterns and Frameworks 25

# Table of Tables

[Table 1: Actors Summary 2](#_Toc468735897)

[Table 2: Artifacts and Information Summary 3](#_Toc468735898)

[Table 3: Process Description: Log in 4](#_Toc468735899)

[Table 4: Typical Course of Action: Login successful 5](#_Toc468735900)

[Table 5: Alternate Course of Action: Login failed 5](#_Toc468735901)

[Table 6: Typical Course of Action: Login With Facebook 5](#_Toc468735902)

[Table 7: Process Description: Log out 5](#_Toc468735903)

[Table 8: Typical Course of Action: Logout successful 5](#_Toc468735904)

[Table 9: Process Description: Upload High Resolution Picture 6](#_Toc468735905)

[Table 10: Typical Course of Action: upload successful 6](#_Toc468735906)

[Table 11: Alternate Course of Action: upload failed 6](#_Toc468735907)

[Table 12: Process Description: Delete Picture 7](#_Toc468735908)

[Table 13: Typical Course of Action: confirm delete 7](#_Toc468735909)

[Table 14: Alternate Course of Action: cancel delete 7](#_Toc468735910)

[Table 15: Process Description: Upload Multiple Pictures 8](#_Toc468735911)

[Table 16: Typical Course of Action: upload successful 8](#_Toc468735912)

[Table 17: Alternate Course of Action: upload failed 8](#_Toc468735913)

[Table 18: Process Description: Like Picture 9](#_Toc468735914)

[Table 19: Typical Course of Action: like picture 9](#_Toc468735915)

[Table 20: Process Description: Sort Pictures 9](#_Toc468735916)

[Table 21: Typical Course of Action: sort by number of like 10](#_Toc468735917)

[Table 22: Alternate Course of Action: sort by uploading time 10](#_Toc468735918)

[Table 23: Process Description: Download Multiple Pictures 10](#_Toc468735919)

[Table 24: Typical Course of Action: download successful 11](#_Toc468735920)

[Table 25: Alternate Course of Action: download failed 11](#_Toc468735921)

[Table 26: Process Description: Report Pictures 11](#_Toc468735922)

[Table 27: Typical Course of Action: submit report 11](#_Toc468735923)

[Table 28: Alternate Course of Action: download failed 12](#_Toc468735924)

[Table 29: Process Description: Comment Picture 12](#_Toc468735925)

[Table 30: Typical Course of Action: Comment picture 12](#_Toc468735926)

[Table 31: Process Description: Search Event’s Pictures 12](#_Toc468735927)

[Table 32: Typical Course of Action: view event pictures 13](#_Toc468735928)

[Table 33: Process Description: Create Private Event 13](#_Toc468735929)

[Table 34: Typical Course of Action: create private event successful 13](#_Toc468735930)

[Table 35: Typical Course of Action: create private event failed 14](#_Toc468735931)

[Table 36: Process Description: Create Public Event 14](#_Toc468735932)

[Table 37: Typical Course of Action: create public event successful 14](#_Toc468735933)

[Table 38: Typical Course of Action: create public event failed 15](#_Toc468735934)

[Table 39: Process Description: Delete Event 15](#_Toc468735935)

[Table 40: Typical Course of Action: Delete Event Submit 15](#_Toc468735936)

[Table 41: Typical Course of Action: Delete Event Cancel 16](#_Toc468735937)

[Table 42: Hardware Component Description 19](#_Toc468735938)

[Table 43: Software Component Description 20](#_Toc468735939)

[Table 44: Design Class Description 20](#_Toc468735940)

[Table 45: Architectural Styles, Patterns, and Frameworks 25](#_Toc468735941)

# Table of Figures

[Figure 1: System Context Diagram 2](#_Toc468737605)

[Figure 2: Artifacts and Information Diagram 3](#_Toc468737606)

[Figure 3: Process Diagram 4](#_Toc468737607)

[Figure 4: Hardware Component Class Diagram 18](#_Toc468737608)

[Figure 5: Software Component Class Diagram 19](#_Toc468737609)

[Figure 6: Deployment Diagram 19](#_Toc468737610)

[Figure 7: Design Class Diagram 20](#_Toc468737611)

[Figure 8: Create Event Sequence Diagram 22](#_Toc468737612)

[Figure 9: Upload Image Sequence Diagram 23](#_Toc468737613)

[Figure 10: Download Image Sequence Diagram 23](#_Toc468737614)

[Figure 11: Search Event Sequence Diagram 24](#_Toc468737615)

### Introduction

#### Purpose of the SSAD

Provide the design of the use cases of ShareWeb. Describe the details and give the guidelines of the system for the implementation team.

#### Status of the SSAD

Current version is the final version. It contains the whole part of SSAD and some high level system analysis.

### System Analysis

#### System Analysis Overview

The purpose of the ShareWeb is to accomplish some features that ShareApp doesn’t have. Such as download many large size pictures for an event to user’s computer, and upload high resolution pictures taken by professional camera for an event from user’s computer.

##### System Context

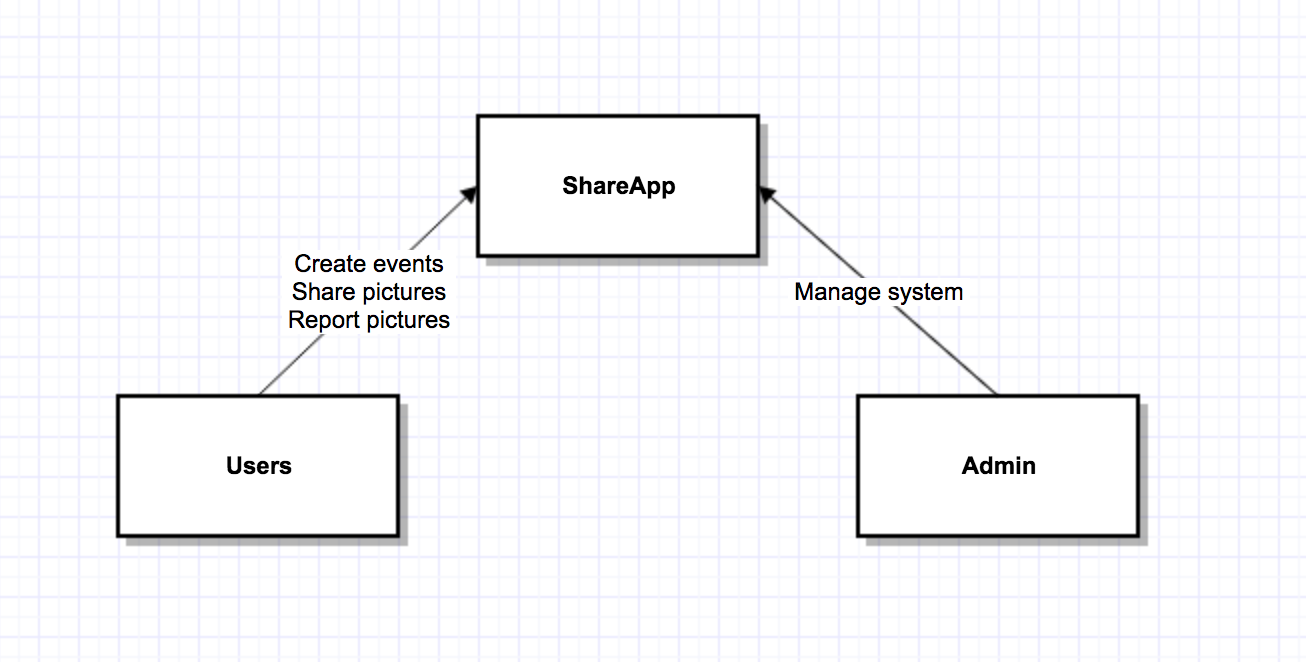


Figure : System Context Diagram

Since the Admin part have already been implemented by previous team, we’ll handle the user part.

Table : Actors Summary

| **Actor** | **Description** | **Responsibilities** |
| --- | --- | --- |
| User | People who use the ShareWeb | Create/search events, delete events, upload/download pictures for events. |
| Admin | People maintain the system | System configuration, delete reported pictures. |

##### Artifacts & Information

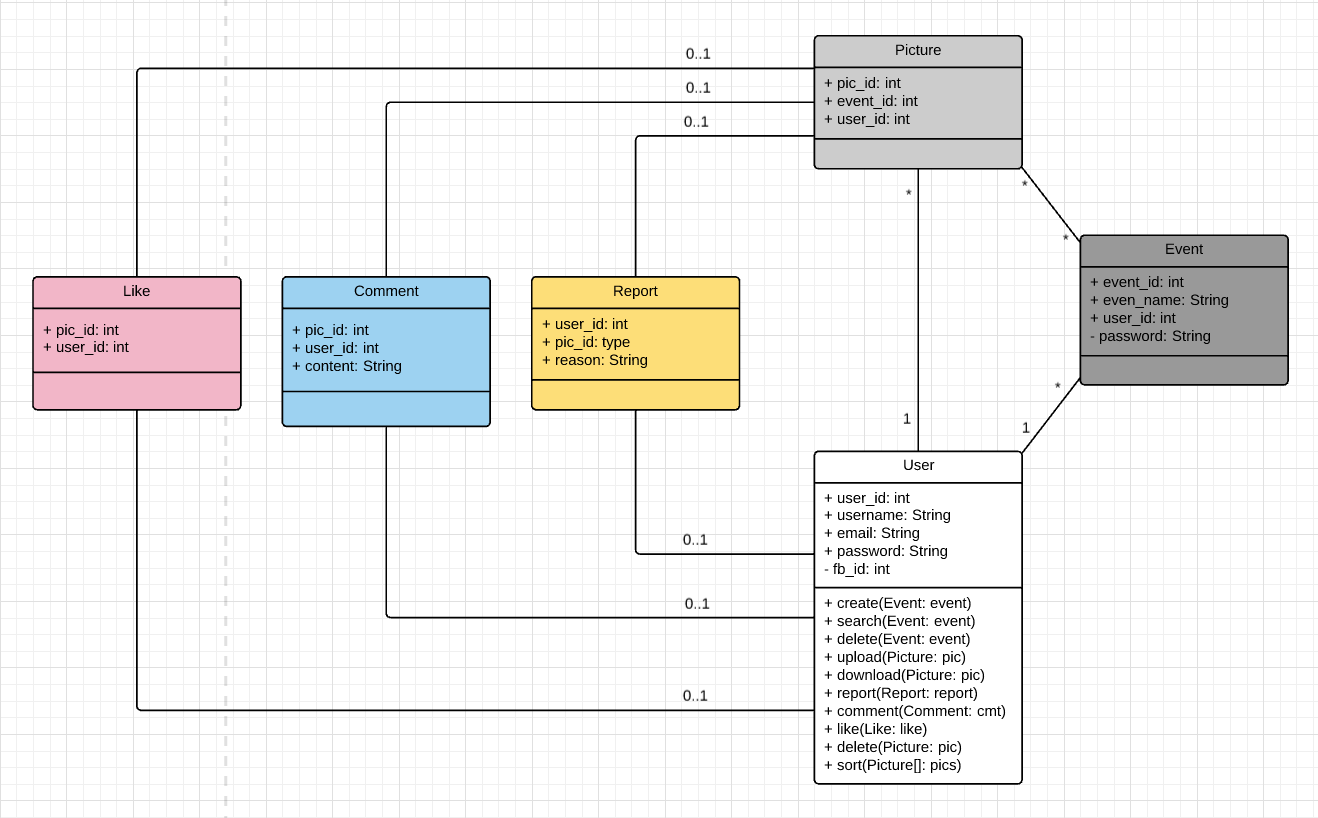


Figure : Artifacts and Information Diagram

Table : Artifacts and Information Summary

|  |  |
| --- | --- |
| **Artifact** | **Purpose** |
| Picture | Contains picture itself and information about the picture. |
| Event | Contains the name of the event, creator of the event, and the password if it is a private event. |
| User | Contains username, email, password for each user. |
| Report | For reporting pictures, contains the picture id, user id and the reason of the report. |
| Comment | Contains information about which user comment what on which pictures. |
| Like | Contains information about which user like which pictures. |

##### Behavior

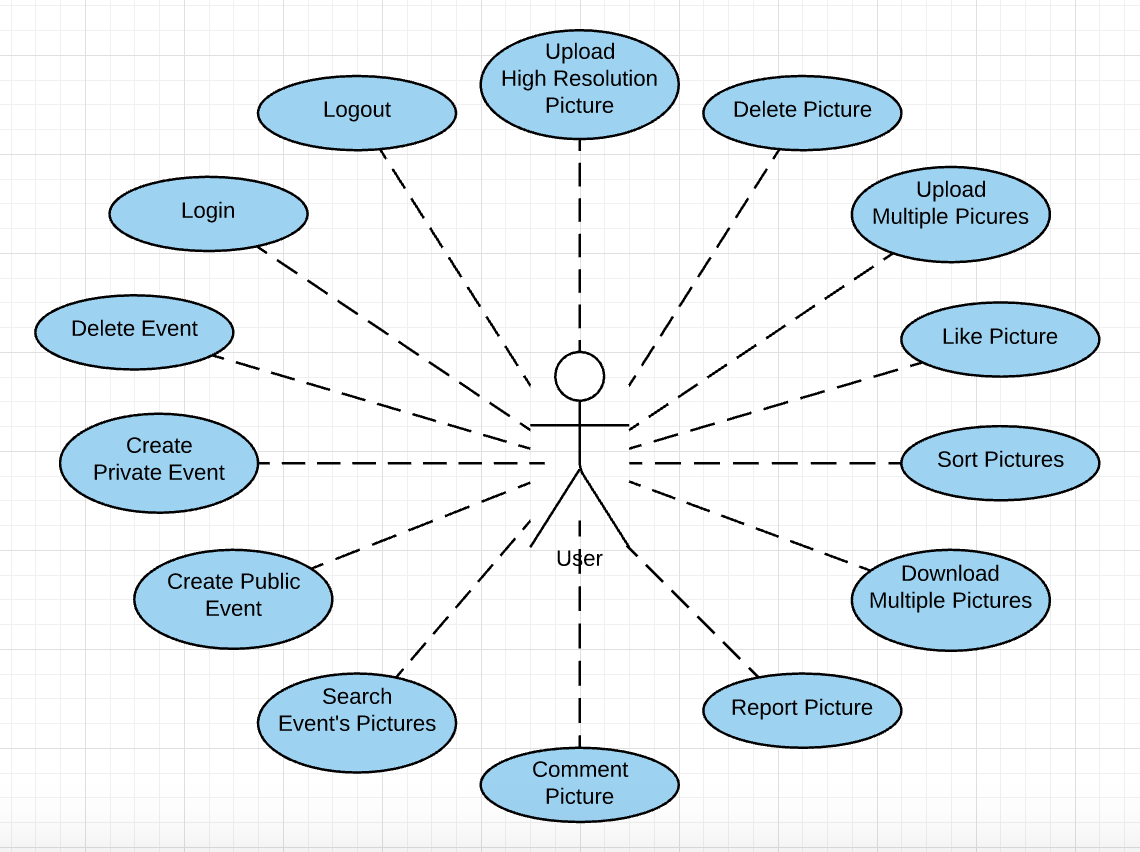


Figure : Process Diagram

###### Authentication

Log in

Table : Process Description: Log in

|  |  |
| --- | --- |
| **Identifier** | UC-1: Log In |
| **Purpose** | Authorize the user and log in to the system. |
| **Requirements** | WC\_3957: As a user, I can log in/out, so that I can use the system.  WC\_3961: As a user, I can login with Facebook account, so that I can login more easier. |
| **Development Risks** | No. |
| **Pre-conditions** | The user is on login page. The user has an email account or Facebook account. |
| **Post-conditions** | The user is authorized and get access to the system. Otherwise the access is denied. |

Table : Typical Course of Action: Login successful

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Enter username and password. |  |
| **2** | Click login button. |  |
| **3** |  | Verify and authorize the user. |
| **4** |  | Redirect the user to Homepage. |

Table : Alternate Course of Action: Login failed

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Enter username and password. |  |
| **2** | Click login button. |  |
| **3** |  | Verify and authorize the user. |
| **4** |  | Show fail message to the user. |

Table : Typical Course of Action: Login With Facebook

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User clicks “login with Facebook” button. |  |
| **2** |  | System directs user to Facebook page and asks the permission |
| **4** |  | Redirect the user to Homepage. |

* + - * 1. Log out

Table : Process Description: Log out

|  |  |
| --- | --- |
| **Identifier** | UC-2: Log Out |
| **Purpose** | Logout of the system and remove the user’s session. |
| **Requirements** | WC\_3957: As a user, I can log in/out, so that I can use the system |
| **Development Risks** | No. |
| **Pre-conditions** | The user is logged in. |
| **Post-conditions** | The user is not authorized to the system, and the session has been removed. |

Table : Typical Course of Action: Logout successful

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Enter username and password. |  |
| **2** | Click logout button. |  |
| **3** |  | Verify and authorize the user. |
| **4** |  | Redirect the user to Homepage. |

###### Picture related behaviors

Upload High Resolution Picture

Table : Process Description: Upload High Resolution Picture

|  |  |
| --- | --- |
| **Identifier** | UC-3: Upload Picture |
| **Purpose** | Allowing user to add pictures for an event. |
| **Requirements** | WC\_4057: As a user, I upload high resolution pictures for an event. |
| **Development Risks** | The size of the picture is too large for uploading. |
| **Pre-conditions** | The user is logged in. |
| **Post-conditions** | A picture is added to the system by the user. |

Table : Typical Course of Action: upload successful

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User click upload picture for event button. |  |
| **2** |  | System opens up local file library |
| **3** | User choose a picture to upload for the event. |  |
| **4** |  | System shows the picture and text field for caption and event to post on. |
| **5** | User fill in the text fields and clicks submit button |  |
| **6** |  | System store the picture for the event with the caption. |

Table : Alternate Course of Action: upload failed

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User click upload picture for event button. |  |
| **2** |  | System opens up local file library |
| **3** | User choose a picture to upload for the event. |  |
| **4** |  | System shows the picture and text field for caption and event to post on. |
| **5** | User fill in the text fields and clicks submit button |  |
| **6** |  | System cannot store the picture and the information to the database. |
| **7** |  | System shows an upload failed message to the user. |

Delete Picture

Table : Process Description: Delete Picture

|  |  |
| --- | --- |
| **Identifier** | UC-4: Delete Picture |
| **Purpose** | Allowing user to delete the picture he/she uploaded. |
| **Requirements** | WC\_3907: As a user, I can delete the picture I upload, so that I can delete the picture I don’t want to share anymore. |
| **Development Risks** | None. |
| **Pre-conditions** | The user is logged in. |
| **Post-conditions** | A picture is removed from the system. |

Table : Typical Course of Action: confirm delete

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User chooses the picture he/she uploaded. |  |
| **2** | User clicks the Delete button. | System shows a warning to the user to confirm the Delete action. |
| **3** | User choose Delete. |  |
| **4** |  | System delete the picture and close the warning. |

Table : Alternate Course of Action: cancel delete

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User chooses the picture he/she uploaded. |  |
| **2** | User clicks the Delete button. | System shows a warning to the user to confirm the Delete action. |
| **3** | User choose Cancel. |  |
| **4** |  | System close the warning without deleting the picture. |

Upload Multiple Pictures

Table : Process Description: Upload Multiple Pictures

|  |  |
| --- | --- |
| **Identifier** | UC-5: Upload Multiple Pictures |
| **Purpose** | Allowing user to upload multiple pictures for an event at one time. |
| **Requirements** | WC\_3990: As a user, I can upload multiple pictures at one time, so that I can upload pictures with less time. |
| **Development Risks** | None. |
| **Pre-conditions** | The user is logged in. |
| **Post-conditions** | Multiple pictures are added to an event by the user. |

Table : Typical Course of Action: upload successful

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User click upload multiple pictures for event button. |  |
| **2** |  | System opens up local file library |
| **3** | User choose multiple pictures to upload for the event. |  |
| **4** |  | System shows the pictures and text field for caption and event to post on. |
| **5** | User fill in the text fields and clicks submit button |  |
| **6** |  | System store the pictures for the event with the caption. |

Table : Alternate Course of Action: upload failed

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User click upload picture for event button. |  |
| **2** |  | System opens up local file library |
| **3** | User choose multiple pictures to upload for the event. |  |
| **4** |  | System shows the pictures and text field for caption and event to post on. |
| **5** | User fill in the text fields and clicks submit button |  |
| **6** |  | System cannot store some pictures and the information to the database. |
| **7** |  | System shows an upload failed message with the failed pictures to the user. |

Like Picture

Table : Process Description: Like Picture

|  |  |
| --- | --- |
| **Identifier** | UC-6: Like Picture |
| **Purpose** | Allowing user to like a picture. |
| **Requirements** | WC\_3908: As a user, I can like a picture. |
| **Development Risks** | None. |
| **Pre-conditions** | The user is logged in. The user hasn’t liked the picture. |
| **Post-conditions** | The picture is liked by the user. |

Table : Typical Course of Action: like picture

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User click on the heart icon for a picture. |  |
| **2** |  | System update the picture’s like information. |

Sort Pictures

Table : Process Description: Sort Pictures

|  |  |
| --- | --- |
| **Identifier** | UC-7: Sort Pictures |
| **Purpose** | Allowing user to sort pictures by popularity or date. |
| **Requirements** | WC\_4090: As a user, I can sort picture based on the picture popularity, so that I can review pictures according to their popularity.  WC\_4091: As a user, I can sort pictures in an event based on the uploaded time, so that I can review pictures according to their upload time. |
| **Development Risks** | None. |
| **Pre-conditions** | The user is logged in. |
| **Post-conditions** | The pictures presented to the user are sorted by number of Likes or date. |

Table : Typical Course of Action: sort by number of like

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User click the filter icon |  |
| **2** |  | System shows options of sorting by popularity or date. |
| **3** | User choose sort by popularity |  |
| **4** |  | System sorts the pictures by number of Likes. |

Table : Alternate Course of Action: sort by uploading time

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User click the filter icon |  |
| **2** |  | System shows options of sorting by popularity or date. |
| **3** | User choose sort by date |  |
| **4** |  | System sorts the pictures by uploading time. |

Download Multiple Pictures

Table : Process Description: Download Multiple Pictures

|  |  |
| --- | --- |
| **Identifier** | UC-8: Download Multiple Pictures |
| **Purpose** | Allowing user to download multiple pictures for an event at one time. |
| **Requirements** | WC\_3988: As a user, I can download multiple pictures at one time, so that I can save time. |
| **Development Risks** | None. |
| **Pre-conditions** | The user is logged in. |
| **Post-conditions** | Multiple pictures are saved to local by the user. |

Table : Typical Course of Action: download successful

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User click download multiple pictures button. |  |
| **2** |  | System opens up the multiple choices page for the pictures. |
| **3** | User choose multiple pictures to download. |  |
| **4** |  | System starts the download process. |
| **5** |  | System shows the success message to the user. |

Table : Alternate Course of Action: download failed

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User click download multiple pictures button. |  |
| **2** |  | System opens up the multiple choices page for the pictures. |
| **3** | User choose multiple pictures to download. |  |
| **4** |  | System starts the download process. |
| **5** |  | System shows the fail message to the user. |

Report Picture

Table : Process Description: Report Pictures

|  |  |
| --- | --- |
| **Identifier** | UC-9: Report Pictures |
| **Purpose** | Allowing user to report pictures. |
| **Requirements** | WC\_3909: As a user, I can report a picture, so that I don’t need to see the pictures I don’t want. |
| **Development Risks** | None. |
| **Pre-conditions** | The user is logged in. |
| **Post-conditions** | User reports the specific picture. |

Table : Typical Course of Action: submit report

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User click report button of a picture. |  |
| **2** |  | System shows a conformation dialogue. |
| **3** | User clicks Report button. |  |
| **4** |  | System stores the report. |

Table : Alternate Course of Action: download failed

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User click report button of a picture. |  |
| **2** |  | System shows a conformation dialogue. |
| **3** | User clicks Cancel button. |  |
| **4** |  | System closes the conformation dialogue. |

Comment Picture

Table : Process Description: Comment Picture

|  |  |
| --- | --- |
| **Identifier** | UC-10: Comment Picture |
| **Purpose** | Allowing user to comment on a picture. |
| **Requirements** | WC\_3906: As a user, I can comment a picture, so I can make a comment to the picture owner. |
| **Development Risks** | None. |
| **Pre-conditions** | The user is logged in. |
| **Post-conditions** | The picture is commented by the user. |

Table : Typical Course of Action: Comment picture

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User click on the comment icon for a picture. |  |
| **2** | User write comment and click submit |  |
| **3** |  | System updates the picture’s comment information. |

###### Event related behaviors

Search Event’s Pictures

Table : Process Description: Search Event’s Pictures

|  |  |
| --- | --- |
| **Identifier** | UC-11: Search Event’s Pictures |
| **Purpose** | Allowing user to view pictures for an event. |
| **Requirements** | WC\_4058: As a user, I can search for an event, so that I can view the pictures for the specific event.  WC\_3910: As a user, I can enter the correct password and see pictures in a private event, so that I can see the pictures I’m allowed to see. |
| **Development Risks** | None. |
| **Pre-conditions** | The user is logged in. |
| **Post-conditions** | Users view the pictures in a public/private event. |

Table : Typical Course of Action: view event pictures

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User enters the event name to the text field and clicks Submit button. |  |
| **2** |  | System shows the search result. |
| **3** | User chooses the desired event |  |
| **4** |  | System shows the event pictures to the user. |

Create Private Event

Table : Process Description: Create Private Event

|  |  |
| --- | --- |
| **Identifier** | UC-12: Create Private Event |
| **Purpose** | Allowing user to create a private event for certain people. |
| **Requirements** | WC\_4059: As a user, I can create an event, so that I can upload related pictures to the event. |
| **Development Risks** | None. |
| **Pre-conditions** | The user is logged in. |
| **Post-conditions** | Users created a private event that only invited people can view the pictures of it. |

Table : Typical Course of Action: create private event successful

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User takes a picture and posts it in a private event. |  |
| **2** |  | System shows the private event name and password fields for user to fill. |
| **3** | User fills the private event name field and password field. User clicks the Submit button. |  |
| **4** |  | System checks if the event name is already existed in the database. And if the password is valid. |
| **5** |  | System creates the event |

Table : Typical Course of Action: create private event failed

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User takes a picture and posts it in a private event. |  |
| **2** |  | System shows the private event name and password fields for user to fill. |
| **3** | User fills the private event name field and password field. User clicks the Submit button. |  |
| **4** |  | System checks if the event name is already existed in the database. And if the password is valid |
| **5** |  | System warns the user the event name has already existed or the password is invalid. |

Create Public Event

Table : Process Description: Create Public Event

|  |  |
| --- | --- |
| **Identifier** | UC-13: Create Public Event |
| **Purpose** | Allowing user to create a public event. |
| **Requirements** | WC\_4059: As a user, I can create an event, so that I can upload related pictures to the event. |
| **Development Risks** | None. |
| **Pre-conditions** | The user is logged in. |
| **Post-conditions** | Users created a public event that all other users can access. |

Table : Typical Course of Action: create public event successful

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User takes a picture and posts it in a public event. |  |
| **2** |  | System shows the public event name field for user to fill. |
| **3** | User fills the public event name field. User clicks the Submit button. |  |
| **4** |  | System checks if the event name is already existed in the database. |
| **5** |  | System creates the event |

Table : Typical Course of Action: create public event failed

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User takes a picture and posts it in a public event. |  |
| **2** |  | System shows the public event name and password fields for user to fill. |
| **3** | User fills the public event name field and password field. User clicks the Submit button. |  |
| **4** |  | System checks if the event name is already existed in the database. |
| **5** |  | System warns the user the event name has already existed. |

Delete Event

Table : Process Description: Delete Event

|  |  |
| --- | --- |
| **Identifier** | UC-14: Delete Event |
| **Purpose** | Allowing user to delete its own private event. |
| **Requirements** | WC\_3977: As a user, I can delete my (private) event, so that I can keep my privacy. |
| **Development Risks** | None. |
| **Pre-conditions** | The user is logged in. User is the creator of the private event |
| **Post-conditions** | Users deletes a private event with all associate pictures. |

Table : Typical Course of Action: Delete Event Submit

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User make a delete event request to the system. |  |
| **2** |  | System shows the confirmation warning. |
| **3** | User clicks Delete. |  |
| **4** |  | System closes the warning and deletes the private event with all associated pictures. |

Table : Typical Course of Action: Delete Event Cancel

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User make a delete event request to the system. |  |
| **2** |  | System shows the confirmation warning. |
| **3** | User clicks Delete. |  |
| **4** |  | System closes the warning with the private even remained. |

##### Modes of Operation

ShareWeb doesn’t have multiple modes.

#### System Analysis Rationale

ShareWeb is the website for Share App, user logins to the system using the email registered on Share App or Facebook account. Users can view the pictures for a specific event. Users can create public or private events and post pictures to the events. The system supports uploading high resolution pictures from computers. It allows people to search for a specific event and download multiple pictures of the event to user’s computer at one time.

### Technology-Independent Model

According to the Q&A in Piazza @148, This section is omitted intentionally because we know what technology we are going to use (AngularJS and Bootstrap) at the very beginning of this project, so we jump into the tech-specific section and skip this section.

### Technology-Specific System Design

#### Design Overview

##### System Structure

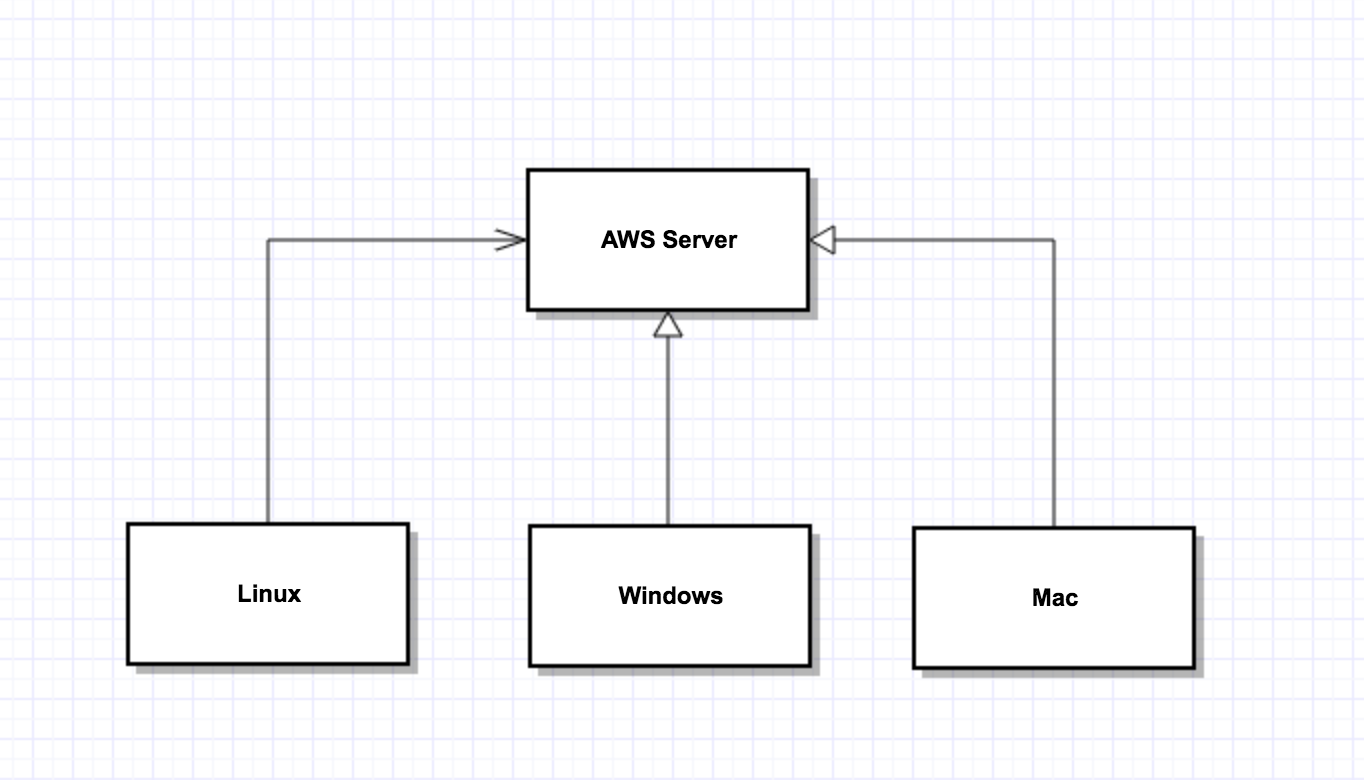
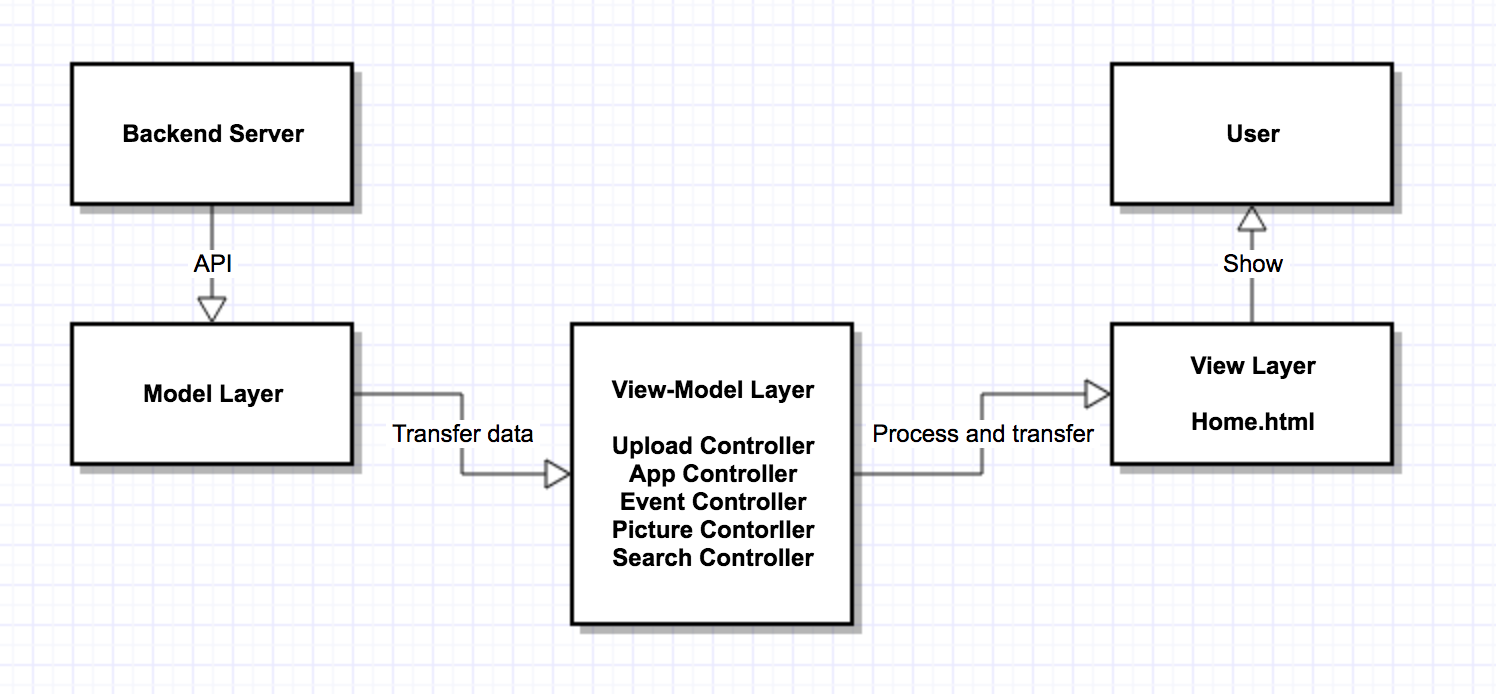


Figure : Hardware Component Class Diagram

Figure : Software Component Class Diagram

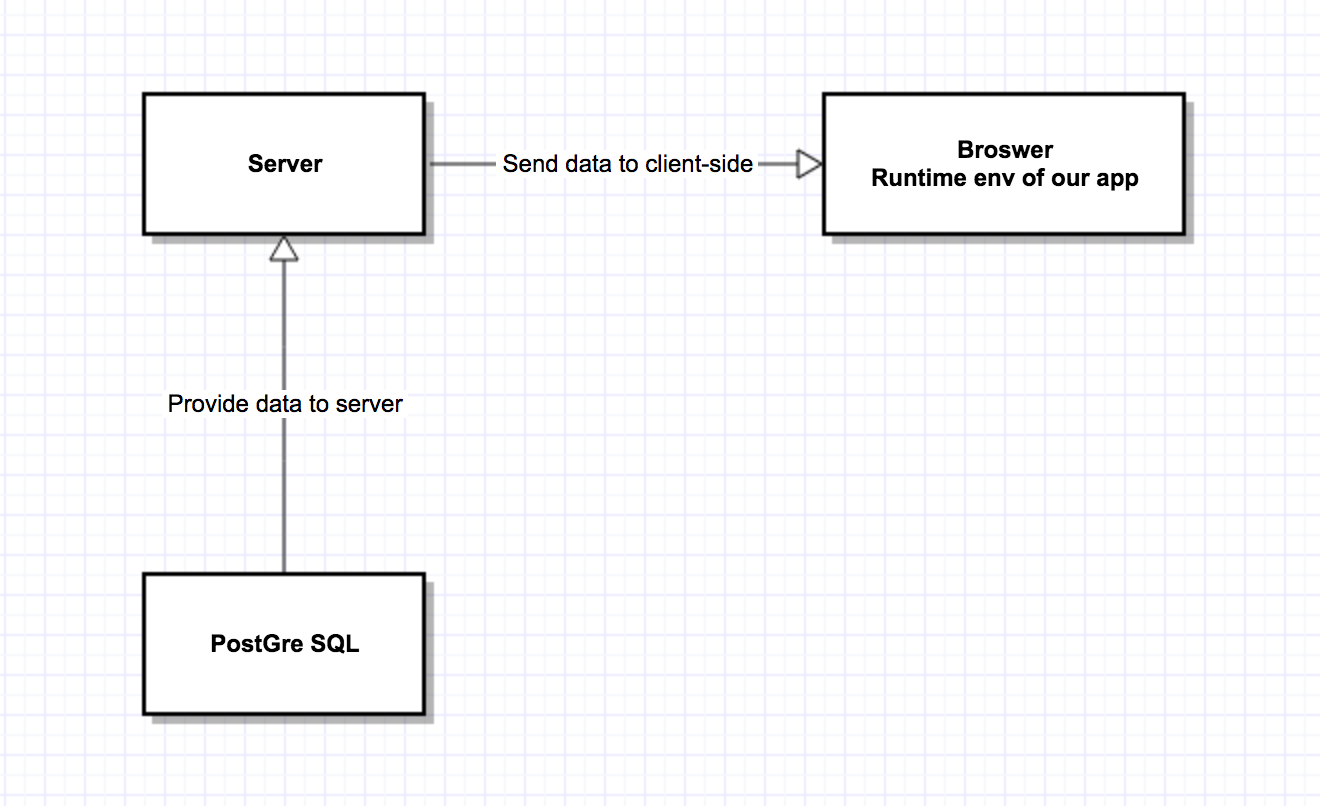
Figure : Deployment Diagram

Table : Hardware Component Description

|  |  |
| --- | --- |
| **Hardware Component** | **Description** |
| Web Server | The hardware our project deployed on, managed by AWS and we control it through AWS dashboard. |
| Linux | Users’ computer with Linux OS |
| Windows | Users’ computer with Windows OS |
| Mac | Users’ computer with Mac OS |

Table : Software Component Description

|  |  |
| --- | --- |
| **Software Component** | **Description** |
| View Layer | Serve as the component that responsible for look, use the fetched data from view-model layer to generate the final page that the users see |
| Model Layer | Serve as the data source for the whole system, responsible for calling backend api and fetch data from it and transfer them to view-model layer |
| View-Model Layer | Serve as a bridge between view and model layer, transfer data from model layer to view layer, control all the behavior within the whole system |

##### Design Classes

###### User-Event-Picture system classes

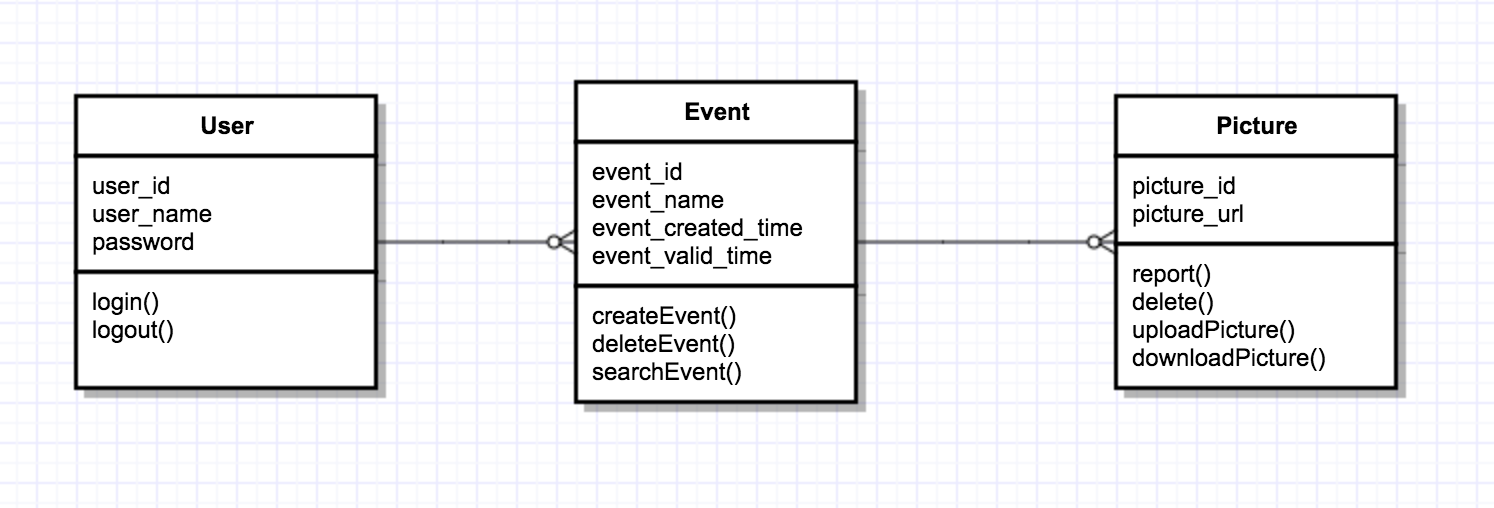
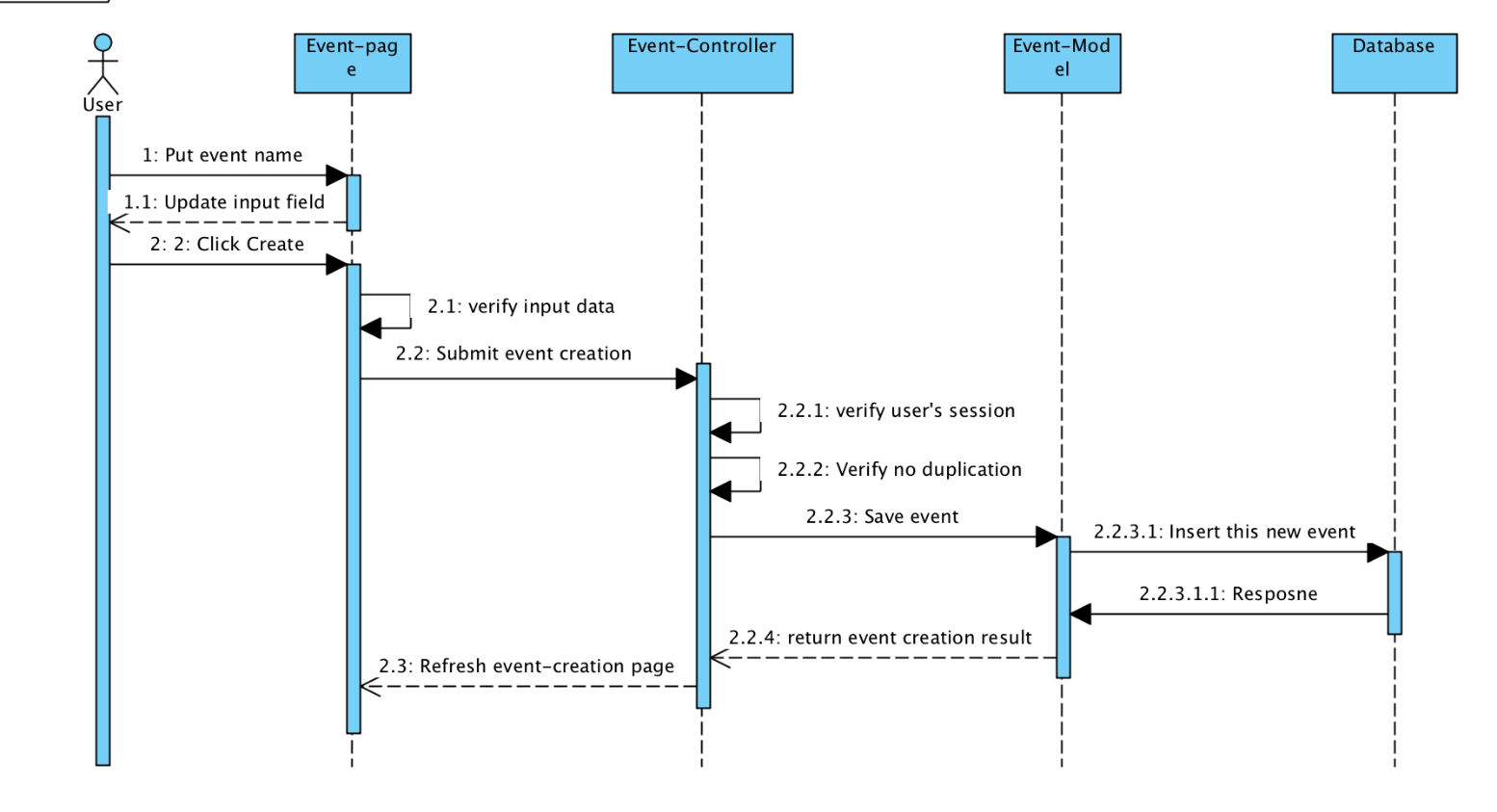
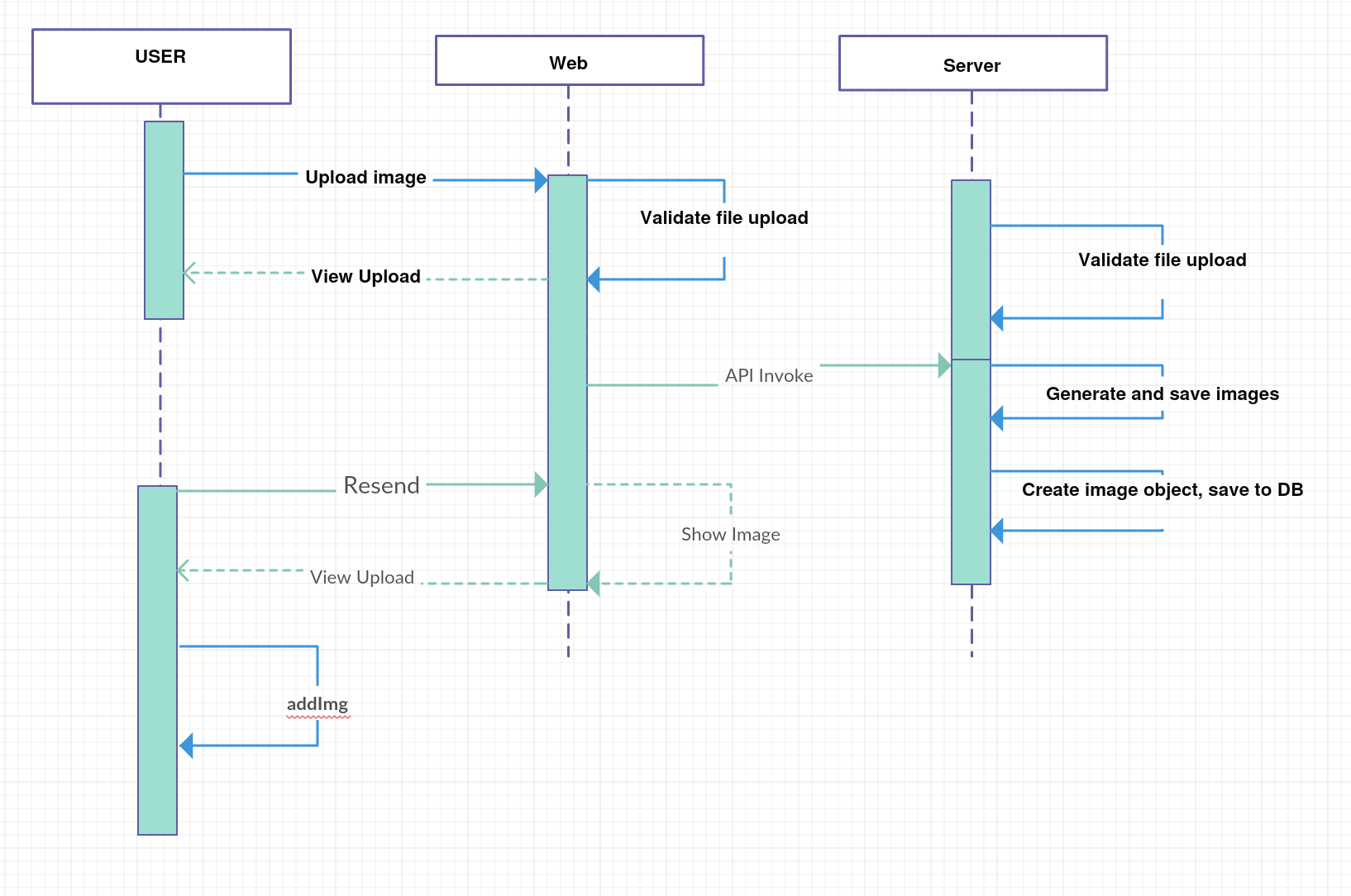
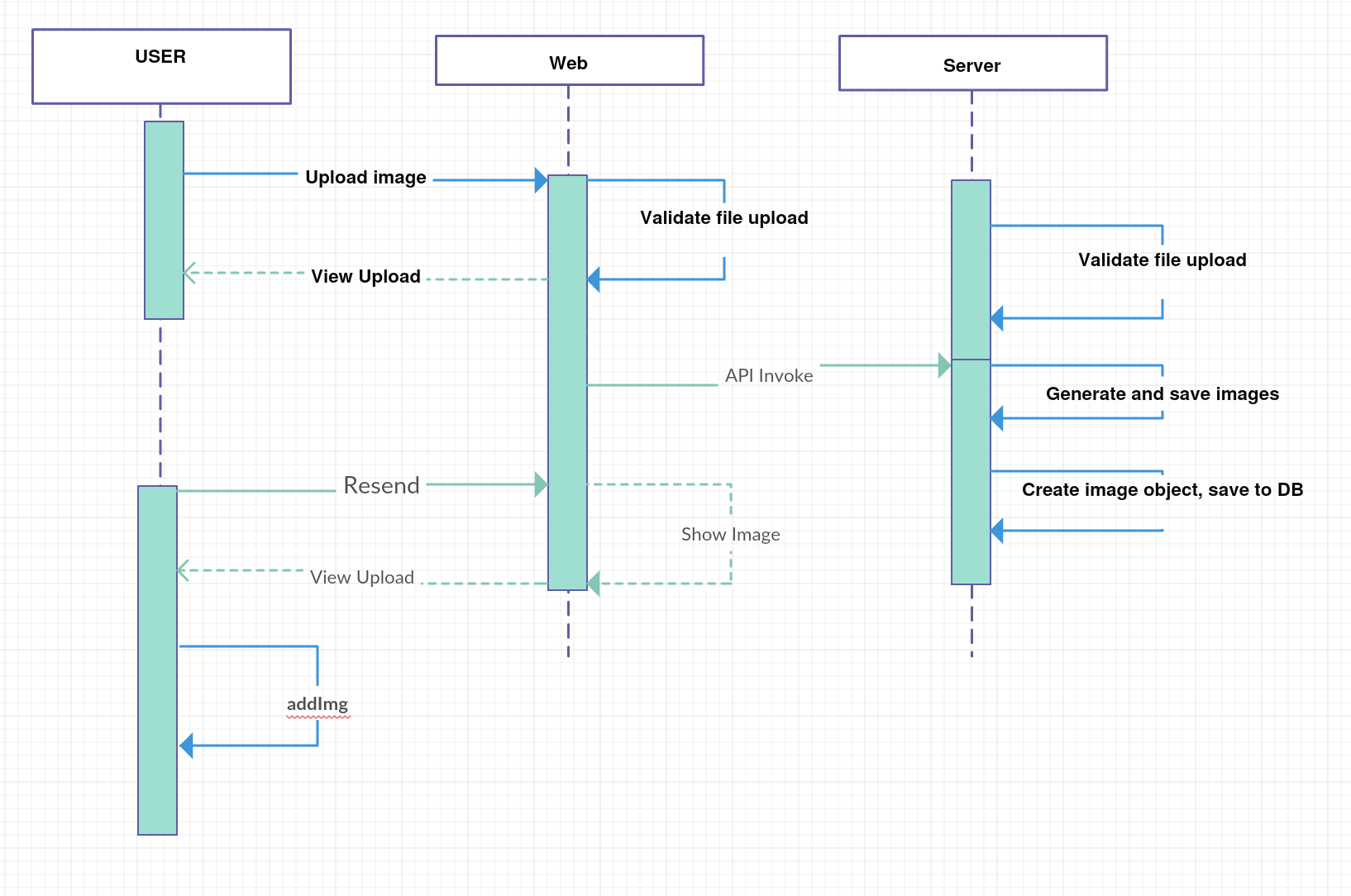
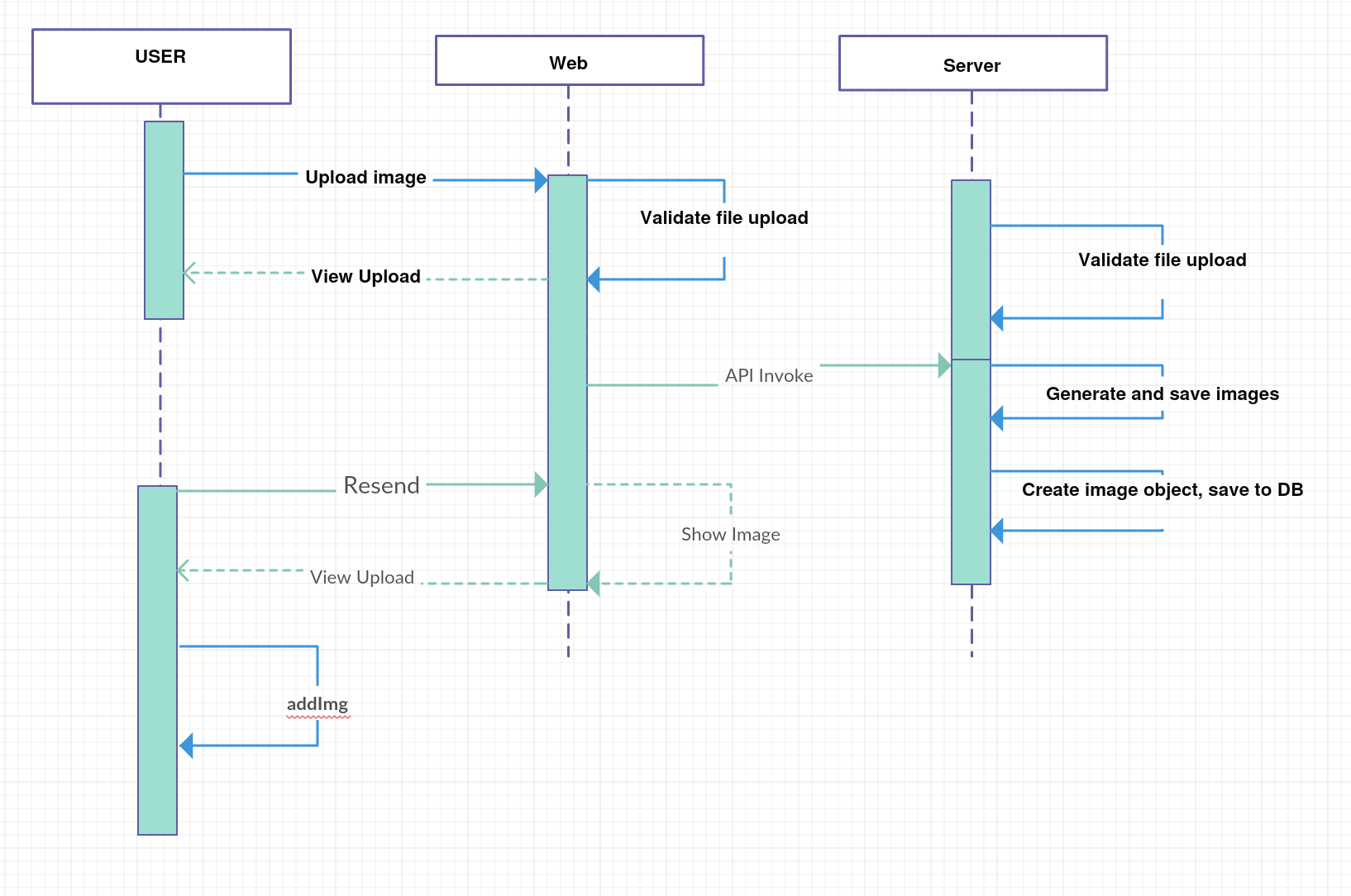
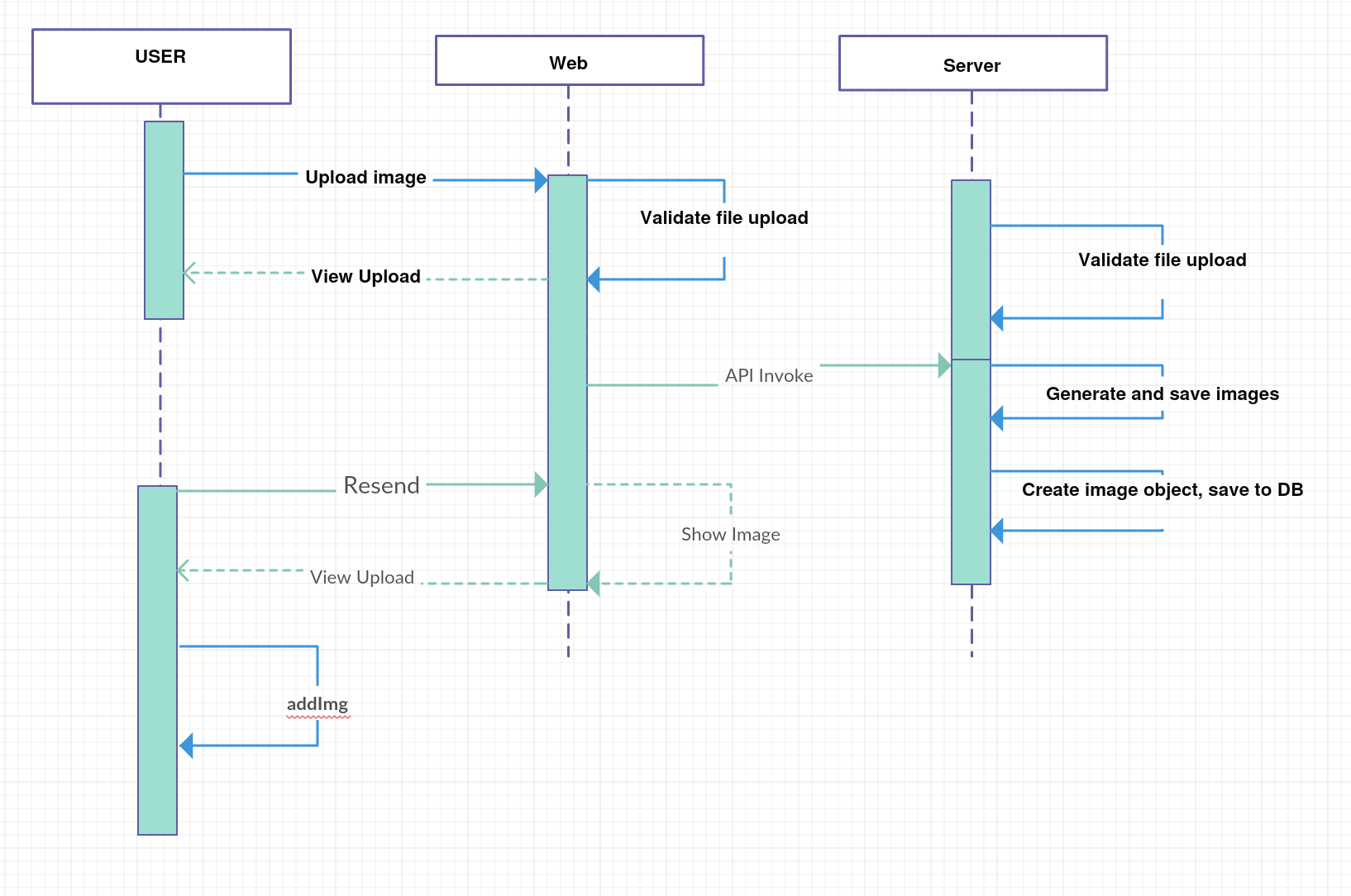
Figure : Design Class Diagram

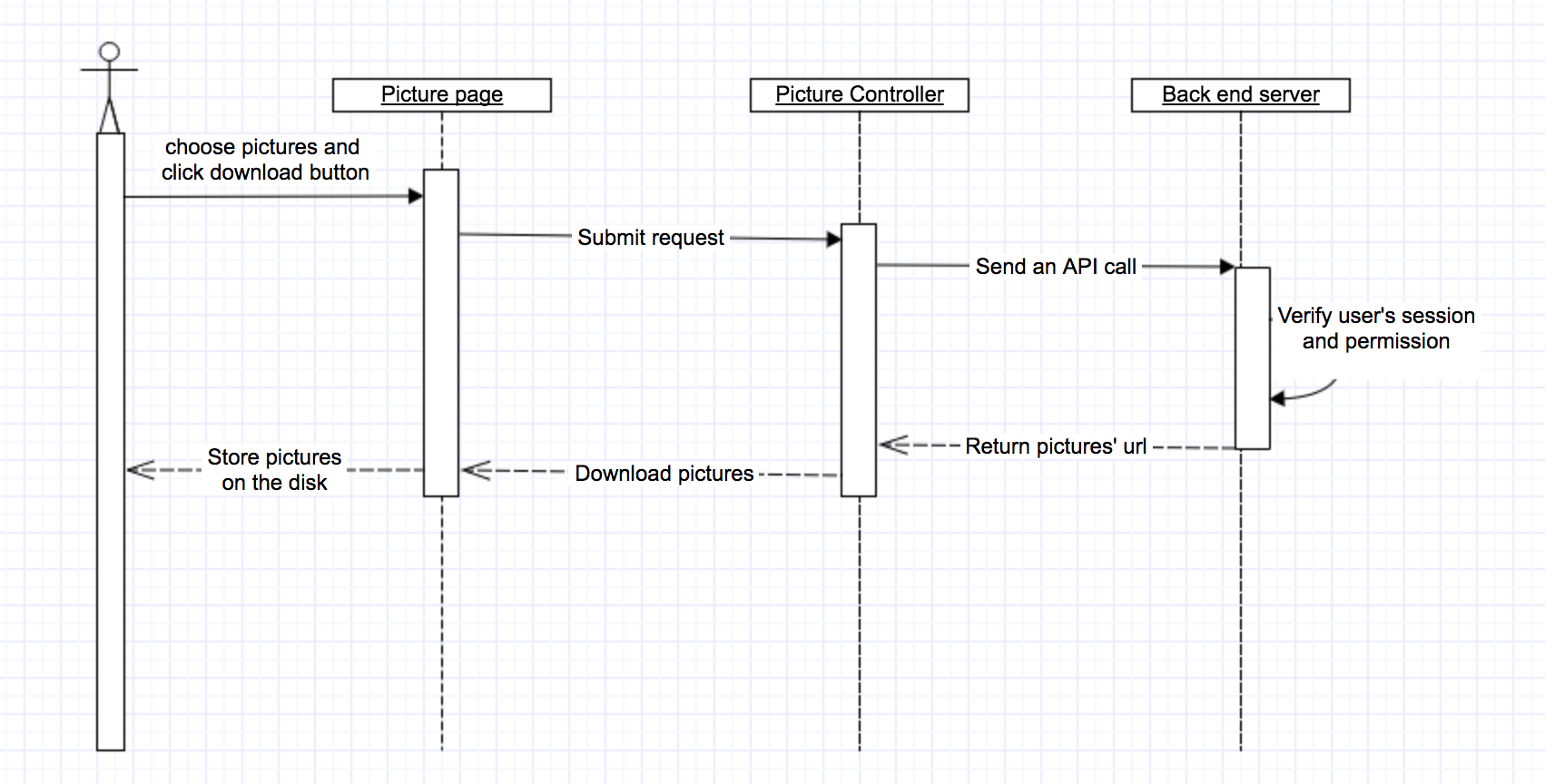
Table : Design Class Description

|  |  |  |
| --- | --- | --- |
| **Class** | **Type** | **Description** |
| User | Entity | A user class that contain id, name and password, used to login and log out |
| Event | Entity | A event class that contains id, name, created\_time and valid\_time, used to create, delete or search |
| Picture | Entity | A picture class that contains id and url, used to report delete upload and download. |

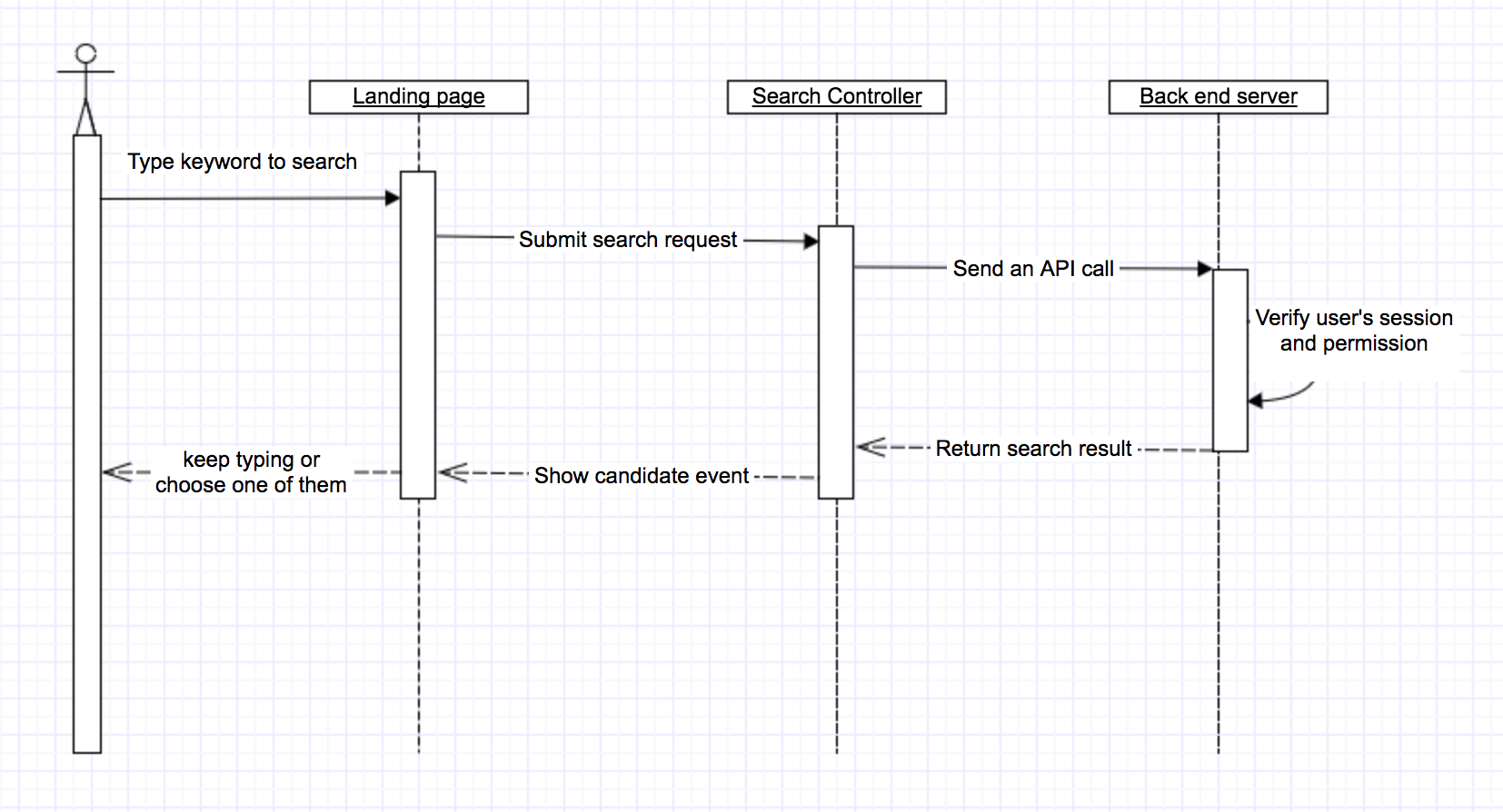
##### Process Realization

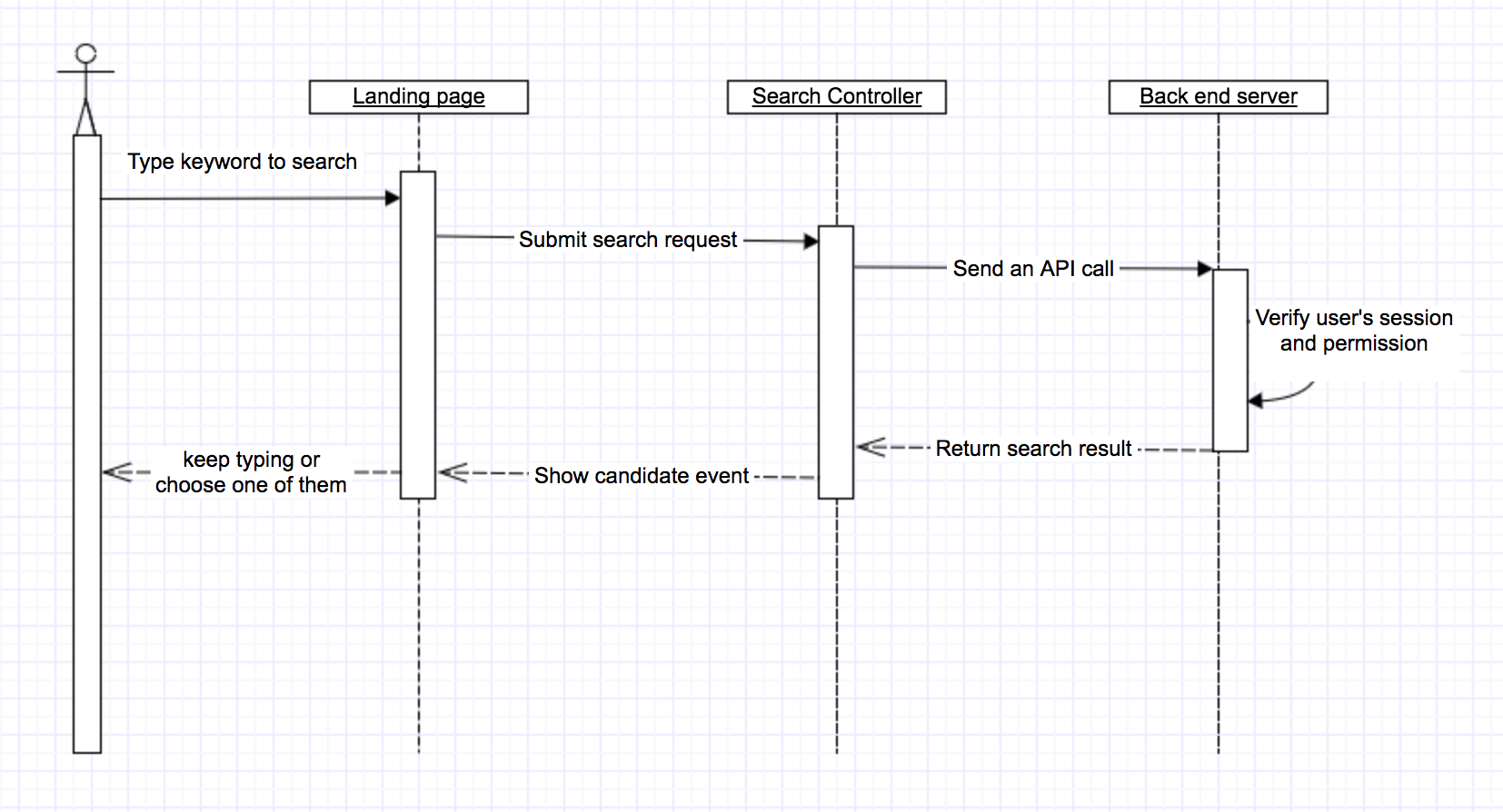
 Figure : Create Event Sequence Diagram



** Figure 9: Upload Image Sequence Diagram**

**Figure 10: Download Image Sequence Diagram**

****

**** **Figure 11: Search Event Sequence Diagram**

#### Design Rationale

Our system follows the design pattern of MVVM which sits behind AngularJS. There are three layers totally.

* View Layer: responsible for the look of page, generate the final page for users.
* Model Layer: responsible for the data used inside page, call api from backend and get data
* View Model Layer: a binding layer bridge view and model, transfer the fetched data to the view layer, control every behavior between view and model.

### Architectural Styles, Patterns and Frameworks

Table : Architectural Styles, Patterns, and Frameworks

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Benefits, Costs, and Limitations** |
| AngularJS | A complete JavaScript-based open-source front-end web application framework | Benefits:  (1) A mature front end framework, provide good solution to web-problem  (2) Secure, fast and well-functional  Limit:  (1) Learning curve is relatively steep |
| Bootstrap | Bootstrap is a free and open-source front-end web framework for designing websites and web applications. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. | Benefit:  (1) Successful solution to front-end view problem  (2) Easily cooperate with other framework like angularJS.  Limits:  (1) The freedom of view is limited and hard to change outside this framework |
| JQuery | jQuery is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML.[ | Benefits:  (1) Provide cross-browser api, write less and achieve more  (2) Basic library for bootstrap and angularjs  Limits:  (1) APIs is limited and still need other framework to work with it together |
| MVVM | Model–view–view-model (MVVM) is a software architectural pattern. | Benefits:  (1) Splits view and model by adding another abstract layer called view-model, so that these two can be developed separately and still work together  (2) Reduce the complexity of the whole system, make it easier to develop and maintain |