

Florida International University
School of Computing and Information Sciences

Software Engineering Focus

Final Deliverable

Streamlining Community Service Process at FIU 1.0

Team Member(s): Kathryn Bello and Ekaterina Gumnova

Project: Streamlining Community Service Process at FIU 1.0

Product Owner(s): Betssy Rodriguez and Lucas Pizzutti

Mentor(s): Allen Varela

Instructor: Masoud Sadjadi

The MIT License (MIT)

Copyright (c) 2016 *Florida International University*

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Abstract

This document presents the information necessary to gain a good understanding of the Streamlining Community Service Process at FIU 1.0 project and its implementation. The product's mission is to connect students to the community by providing a mobile application that simplifies the process of finding and keeping track of volunteering opportunities. This document also describes in detail the features and limitations of the system.

Table of Contents

INTRODUCTION

.....	5
CURRENT SYSTEM	
.....	5
PURPOSE OF NEW SYSTEM	
.....	5

USER STORIES

IMPLEMENTED USER STORIES	
.....	7
PENDING USER STORIES	
.....	10

PROJECT PLAN

HARDWARE AND SOFTWARE RESOURCES	
.....	12
SPRINTS PLAN	
.....	13
<i>Sprint 1</i>	
.....	
..13	
<i>Sprint 2</i>	
.....	
...13	
<i>Sprint 3</i>	
.....	
...14	
<i>Sprint 4</i>	
.....	
...15	
<i>Sprint 5</i>	
.....	
...16	
<i>Sprint 6</i>	
.....	
...17	

<i>Sprint 7</i>	
.....	
...18	
SYSTEM DESIGN	
ARCHITECTURAL PATTERNS	
.....	20
SYSTEM AND SUBSYSTEM DECOMPOSITION	
.....	21
DEPLOYMENT DIAGRAM	
.....	22
DESIGN PATTERNS	
.....	22
SYSTEM VALIDATION	
.....	23
GLOSSARY	
.....	
....	37
APPENDIX	
.....	
.	38
APPENDIX A - UML DIAGRAMS	
.....	38
<i>Static UML Diagrams</i>	
.....	38
<i>Dynamic UML Diagrams</i>	
.....	40
APPENDIX B - USER INTERFACE DESIGN	
.....	52
APPENDIX C - SPRINT REVIEW REPORTS	
.....	69
APPENDIX D - USER MANUALS, INSTALLATION/MAINTENANCE DOCUMENT, SHORTCOMINGS/WISHLIST DOCUMENT AND OTHER DOCUMENTS	
.....	74
REFERENCES	
.....	8
0	

INTRODUCTION

The Streamlining Community Service Process at FIU project emerged from the desire to overcome the obstacles that keep students from engaging in the community. The initial objective was to create a mobile application that provides a more efficient way to find and keep track of community service events available to honors college students. However, this mission expanded to members outside of the FIU community because it was discovered that there is no centralized place to view volunteering events. As the first prototype of Streamlining Community Service Process at FIU, this application will only include the basic functionalities of the system.

Current System

Within the FIU community, the honors college website does not provide a convenient mechanism for keeping track of events that a user is interested in or RSVP's to. There is no tool for sorting through all of the events in the system and there is no mobile application. Outside of the FIU community, there exists mobile applications for volunteering events but they are not intuitive for users. Additionally, these applications do not have standardized requirements for event registration so a lot of the information is vague or confusing.

Purpose of New System

The purpose of the Streamlining Community Service Process at FIU 1.0 project is to streamline the community service search and RSVP process using a mobile application. This application allows users to create, view, filter, locate, and RSVP to volunteering events by combining the functionality of an event scheduler, calendar, and map. Streamlining Community Service Process at FIU 1.0 aims to increase awareness of events in the community and to provide a user friendly online space to bring people together.

USER STORIES

The following section provides the detailed user stories that were implemented in this iteration of the Streamlining Community Service Process 1.0 project. These user stories served as the basis

for the implementation of the project's features. This section also shows the user stories that are to be considered for future development.

Implemented User Stories

676 - Sign up

As a user, I would like to sign up so that I can use the software securely and my data isn't viewed by unregistered users.

Acceptance Criteria:

- A registration form with fields for email, password, firstname, lastname, date of birth, occupation, and location should be provided.
- A button to submit the form should be provided.
- The system should offer the user the option to login (Login button).
- The system must validate the text input by the user.
- The system should automatically login users after a successful registration.

663 - Login

Task: Forgot Password

As a user, I would like to login with my email and password so that I can start using the application.

Acceptance Criteria

- A form should be provided with email and password fields.
- A button should be provided to submit form.
- The system should include "Forgot password" button/functionality.
- The System should offer the user the option to register (Sign up button).
- The system must validate the text input by the user.

671 - Rsvp to Event

Task: Add Event to iPhone Calendar

As a user, I would like to Rsvp to an event so that I can keep track of my volunteering activities.

Acceptance Criteria:

- There should be an Rsvp button.
- The user should receive a confirmation message when the button is selected.

- When the Rsvp button is selected, it should be updated to a Cancel button.
- The system should prompt the user for calendar access after Rsvp is selected (Adds event to user's telephone's calendar).

672 - View Rsvp Calendar

As a user, I would like to view a calendar of volunteer events that I Rsvp'd to so that I can manage and organize my time.

Acceptance Criteria:

- Rsvp'd events should be displayed in a list in chronological order whenever a calendar cell is selected.
- Clicking an event in the list should direct the user to the event details view.
- Each event in the list should have name and time information.
- When the calendar loads, the cell selected should be the current date.

673 - Cancel Rsvp

As a user, I would like to cancel my rsvp so that I can let the organizer know that I am not attending the event.

Acceptance Criteria:

- A cancel button should be provided.
- After the cancel button is selected, an rsvp button should be provided.
- The system does not direct the user to a different view after the cancel button is selected.

664 - View User Profile

As a logged in user, I want to be able to view my profile so that I can view my information.

Acceptance Criteria:

- The user's firstname, lastname, occupation, date of birth, location, and email should be displayed.
- The system should provide an Edit Profile button.

665 - Update User Profile

As a user, I would like to edit my profile so that I can always update my personal information.

Acceptance Criteria:

- A form with optional fields for firstname, lastname, date of birth, occupation, and location should be provided.
- The system should validate text input by the user.
- The system should offer the option to change password. (Manage Password button).
- A “Save” navigation bar button should be provided to submit form.
- A “Cancel” navigation bar button should be provided to return to My Profile without submitting the form.

670 - View Event Details

As a user, I would like to view event details so that I can see if I am interested in that event.

Acceptance Criteria:

- All event information should be displayed (name, date, contact name, contact email, description, attendees list, location).
- There should be a navigation item that the user can select to return to Events or My Events calendar (Back).
- There should be an Rsvp button (User Story 671).
- Social media icons (Twitter, Facebook, Email, LinkedIn) are available to share event (User Story 675).

668 - View Event Calendar

As a User, I want to be able to view a calendar with all of the volunteer events available presented as a list in chronological order, so that I could choose events I am interested in.

Acceptance Criteria:

- A User should be able to view all of the events available displayed in a table view.
- All the events should be displayed in chronological order.
- A User should be able to view each event's name, date, and time.
- A User should be able to click “+” button if he/she wants to add new event to the calendar.
- A User should be able to click on the event if he/she wants to view the event's details.
- A User should be able to see and click on any of the buttons of the tab bar.

677 - View Map

As a User, I want to be able to view a map with all of the volunteer events pinned to the map, so that I could locate those events relative to my location.

Acceptance Criteria:

- A User should be able to view all of the events available pinned on the map.
- A User should be able to view his/her location on the map.
- A User should be able to zoom in and zoom out the map.
- A User should be able to view the search bar in the top left corner of the map.
- A User should be able to click on each pin located on the map.
- A User should be able to see and click on any of the buttons of the tab bar.

667 - Create Event

As a User, I would like to be able to create a volunteering event so that it is added to the events network and displayed in the events calendar.

Acceptance Criteria:

- A User should be able to see placeholder in each field indicating what kind of information is required from him/her.
- A User should be able to fill out all the required fields for a new event to be created.
- A User should be able to pick Starts On and Ends On date & time for the event, using the datepicker.
- A User should not be able to select the end date preceding the start date.
- A User should be able to pick Event Category from the list of the Event Categories available.
- A User should be able to click Save button to add the event to the events calendar.
- A User should be able to click “Back” button to cancel new event creation and return to the events calendar view.
- A User should not be able to save the event if not all of the required fields are completed (a warning should pop up when clicking on the Save button).
- A User should be able to see and click on any of the buttons of the tab bar.

674 - Filter Map results

As a User, I would like to be able to filter the events displayed on the map by the start date of events so that I can see where events, happening on a specific date, are located relative to my location.

Acceptance Criteria:

- A User should be able to view his/her location on the map.
- A User should be able to zoom in and zoom out the map.
- A User should be able to click on the search bar in the top left corner of the map and get the date picker.
- A User should be able to select a date for which he/she wants to map the events.
- A User should be able to click on “Cancel” button on the date picker toolbar to return to the previous view.
- A User should be able to click on “Map” button on the date picker toolbar to map the events that start on the selected date.
- After the user clicks the “Map” button from the date picker, he/she should see the events, that start on the selected date, pinned on the map, and the date that shows the start date for the displayed events in the search bar.
- A User should be able to see and click on any of the buttons of the tab bar.

673 - View Event Information on the Map

As a User, I want to be able to point on the event pin on the map so that I can see event’s name, date, and time.

Acceptance Criteria:

- A User should be able to view his/her location on the map.
- A User should be able to zoom in and zoom out the map.
- A User should be able to point on any pin on the map.
- A User should be able to see event’s information after he/she points on the any pin on the map.
- A User should be able to see and click on any of the buttons of the tab bar.

679 - View the List of the Events Created by the User

As a User, I want to be able to view a list of all the events that I created so that I can edit any of them.

Acceptance Criteria:

- A User should be able to view all of the events he/she created in a chronological order.
- A User should be able to view each event's name, date, and time.
- A User should be able to click the "Back" button if he/she wants to return to the events calendar view.
- A User should be able to click on the event if he/she wants to edit the event's details.
- A User should be able to see and click on any of the buttons of the tab bar.

678 - Edit Event Created by the User

As a User, I want to be able to edit any of the events that I created so that I can update the information if there are any changes.

Acceptance Criteria:

- A User should be able to view all of the details of the selected event.
- A User should be able to edit any of the fields except for the events name.
- A User should be able to click the "Back" button if he/she wants to return to cancel editing the event's details.
- A User should be able to see the "Save" button once he/she starts making changes.
- A User should be able to change start/end time, but not the date.
- A User should be able to see and click on any of the buttons of the tab bar.

Pending User Stories

675 – Share Event on Social Media

As a user, I would like to share an event on social media platforms so that I can promote the volunteer opportunity with all of my friends/followers.

Acceptance Criteria:

- Buttons for Facebook, Twitter and LinkedIn should be provided.
- Event name, description, and link should be shared on user's social media account.

665 – Set Preferences in Profile

As a user, I would like to set my preferences for what event categories I am interested in my profile so that I can only see events according to my set preferences.

Acceptance Criteria:

- A User should be able to select from the list of predefined categories in their profile to set up preferences.

PROJECT PLAN

This section describes the planning that went into the realization of this project. This project incorporated the agile development techniques and as such required the sprints to be planned. These sprint plannings are detailed in the section. This section also describes the components, both software and hardware, chosen for this project.

Hardware and Software Resources

The following is a list of all hardware and software resources that were used in this project:

1. Xcode Software Development Kit.
2. Terminal Version 2.6.1
3. Microsoft PowerPoint for Mac 15.34
4. Mingle - <https://www.thoughtworks.com/mingle/>
5. Google Drive
6. Lucidchart
7. Firebase
8. MacBook Pro (13-inch, Early 2011)
9. CocoaPods

Sprints Plan

Sprint 2

Sprint Planning: 05/26/2017

- #668 - View Event Calendar
- #670 - View Event Details
- #671 - RSVP Event
- #672 - View RSVP Calendar
- #677 - View Map

Sprint 3

Sprint Planning: 06/09/2017

- #680 - Cancel RSVP
- #667 - Create Event

Sprint 4

Sprint Planning: 06/23/2017

- #674 - Filter Map Results
- #664 - View Profile
- #665 - Set Preferences in Profile
- #666 - Update Profile
- #663 - Login
- #676 - Signup
- #679 - View the List of the Events Created by the User
- #678 - Edit Event Created by the User

Sprint 5

Sprint Planning: 07/07/2017

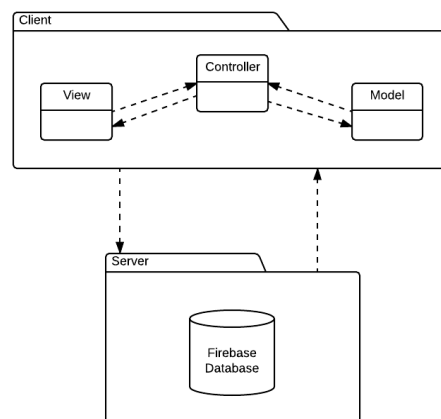
- #675 - Share event on social media

SYSTEM DESIGN

This section contains information on the design decisions that went into this project. The architecture patterns are outlined and explained. The entire system is shown in a package diagram and the subsystems are explained. Finally, the design patterns used in the project are discussed.

Architectural Patterns

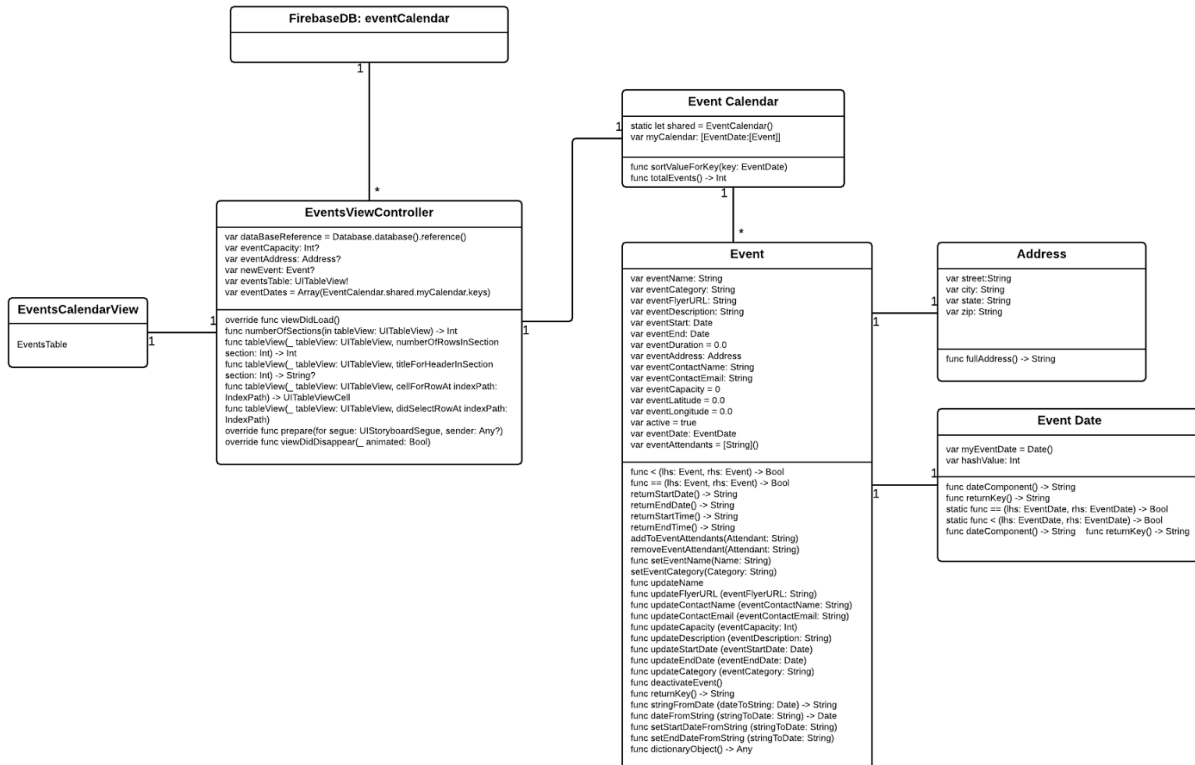
The architectural patterns implemented in our project are Client-server with MVC (Model-View-Controller) for structuring the user interface. In MVC, the communication between the views and the data is provided by the controller classes. For our database server, we implemented Firebase which is a NoSQL, real-time database that saves data as JSON objects. Using Firebase allowed us to keep a 2-tier architecture pattern instead of implementing a traditional 3-tier architecture pattern because Firebase handles all of the server connections.



System and Subsystem Decomposition

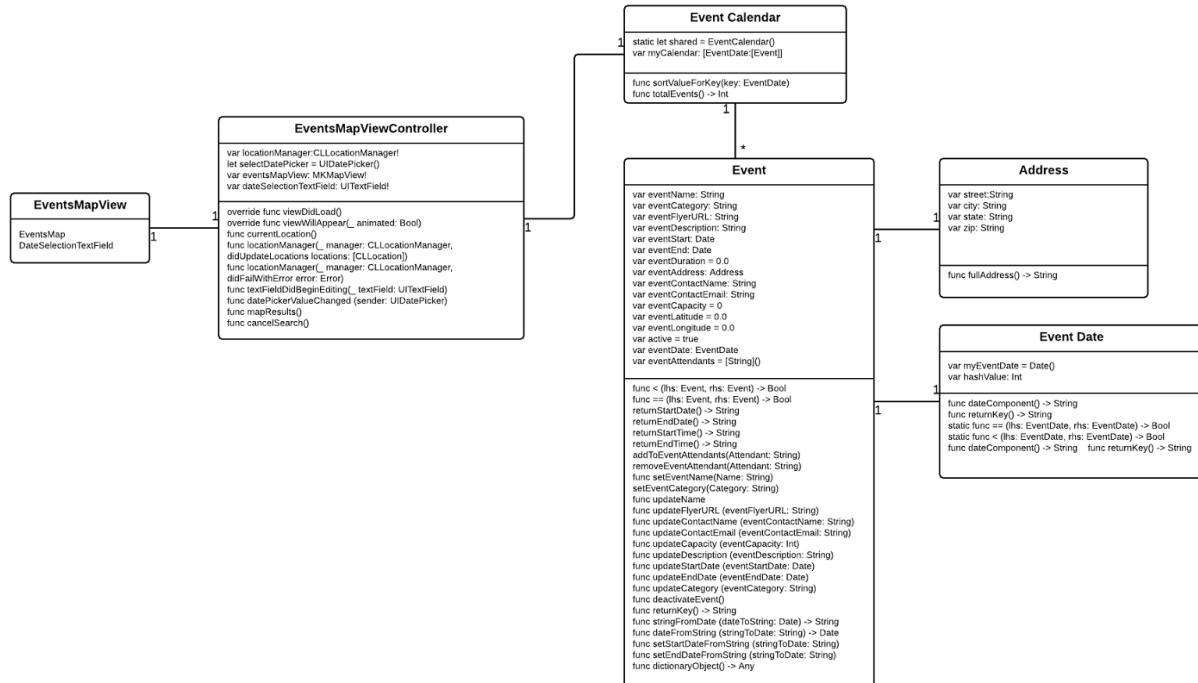
668 - View Event Calendar

Class diagram:



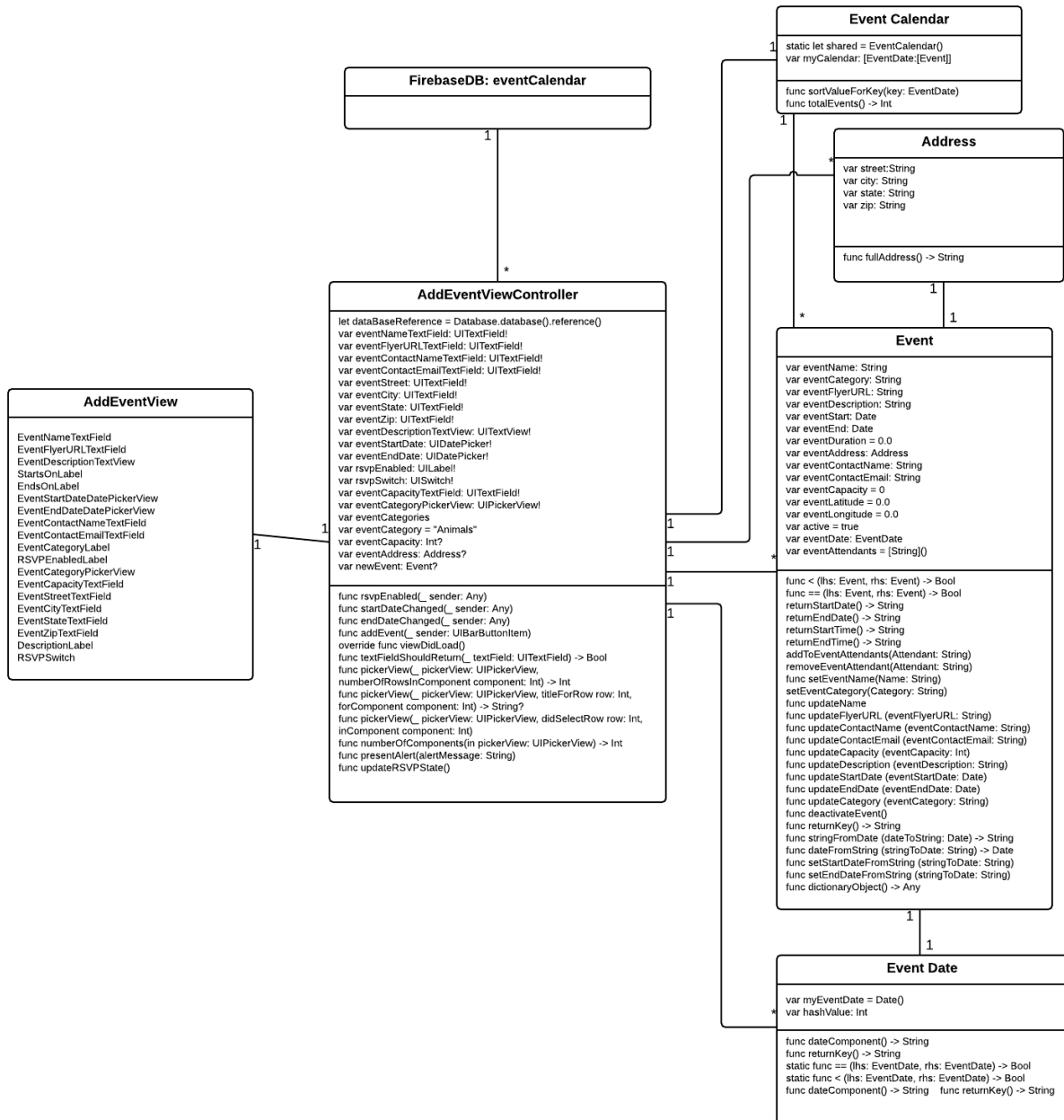
677 - View Map/674 - Filter Map Results/673 - View Event Information on the Map

Class diagram:



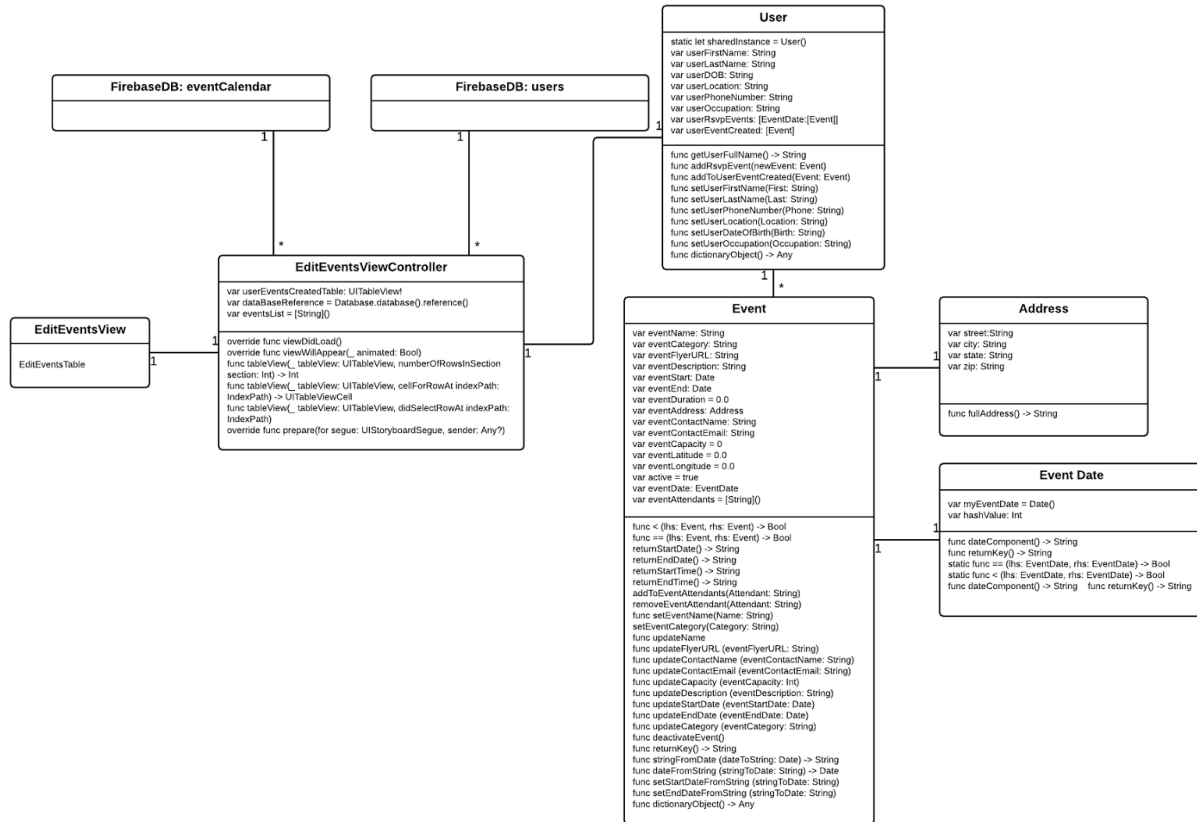
667 - Create Event

Class diagram:

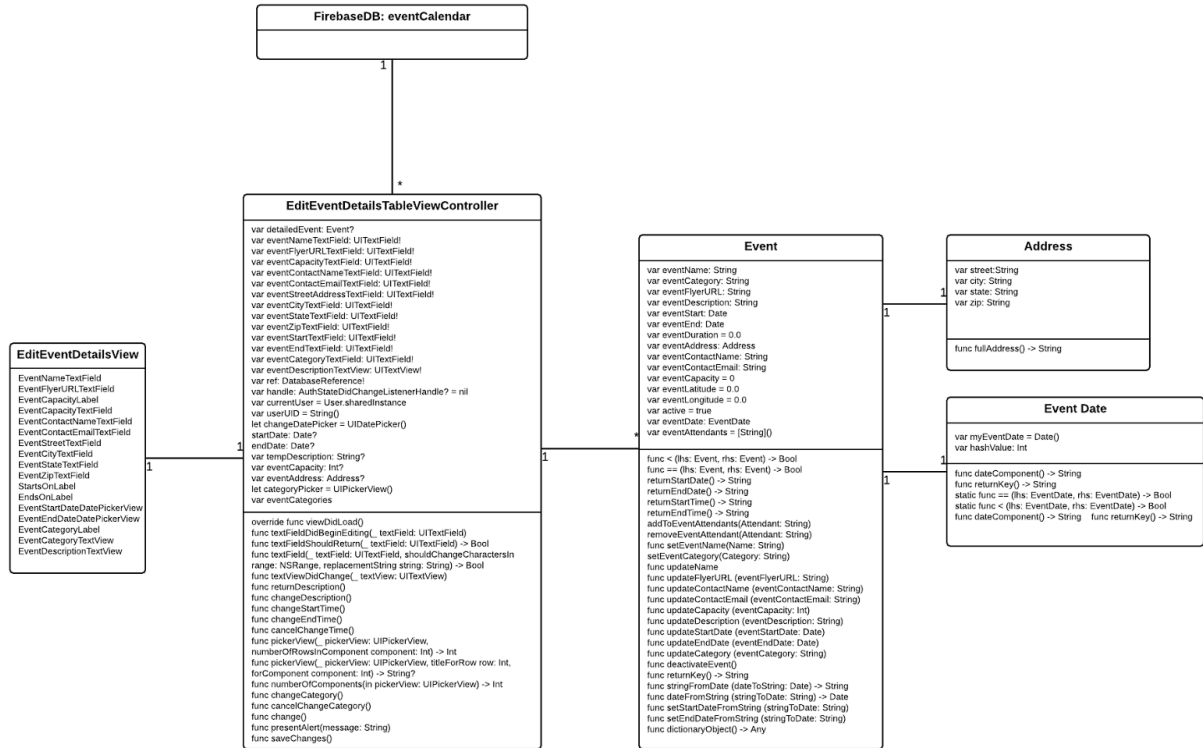


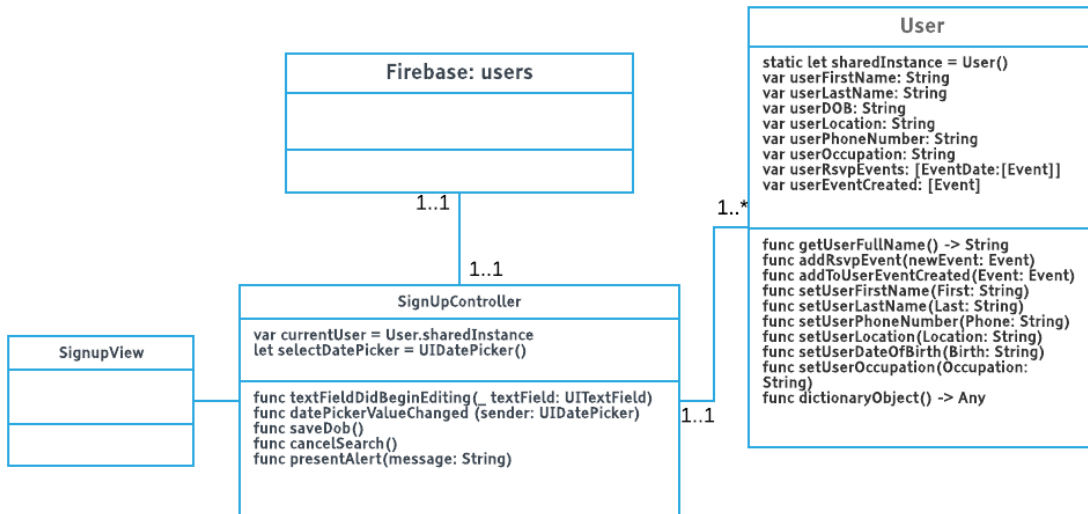
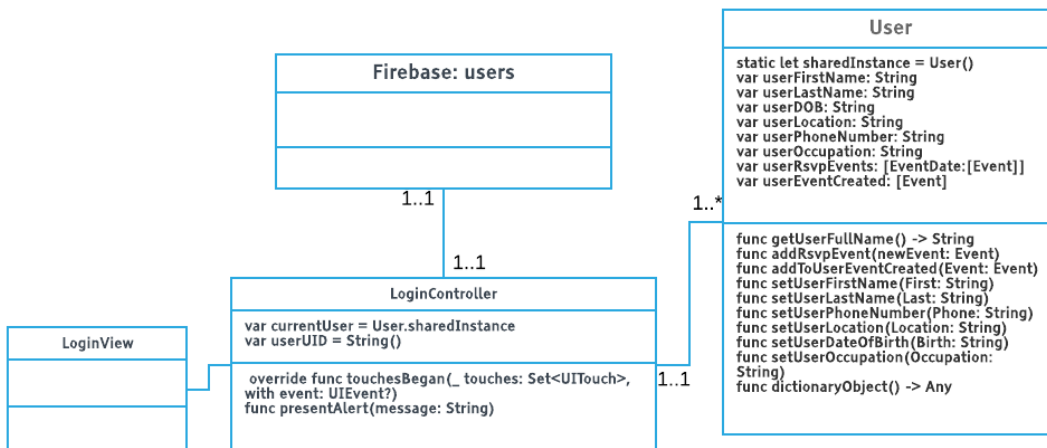
679 - View the List of the Events Created by the User

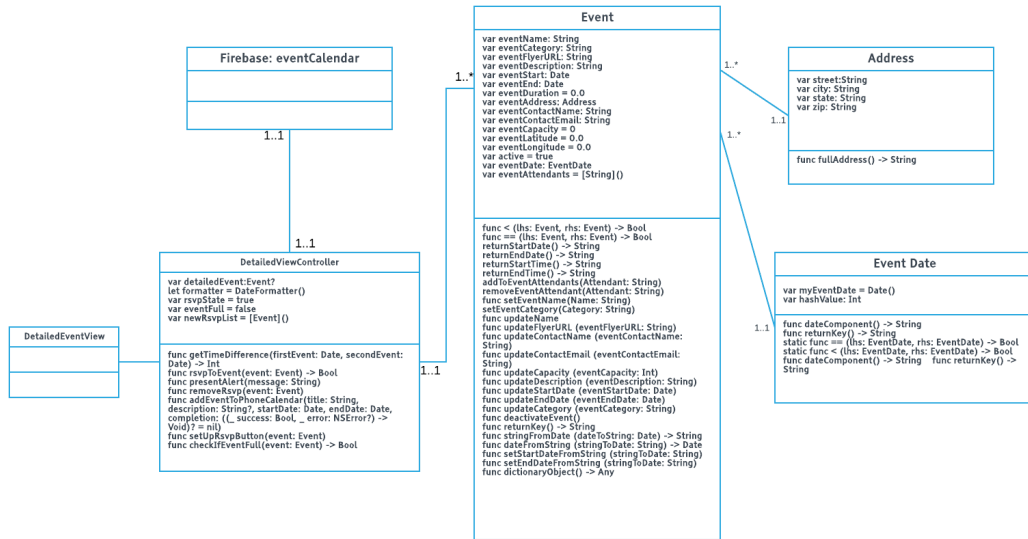
Class diagram:



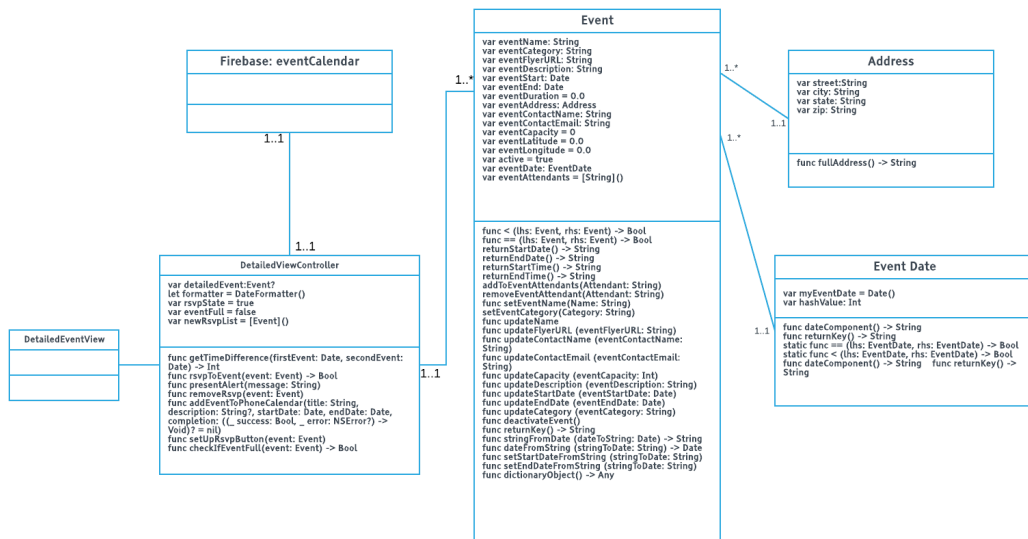
678 - Edit Event Created by the User
Class diagram:



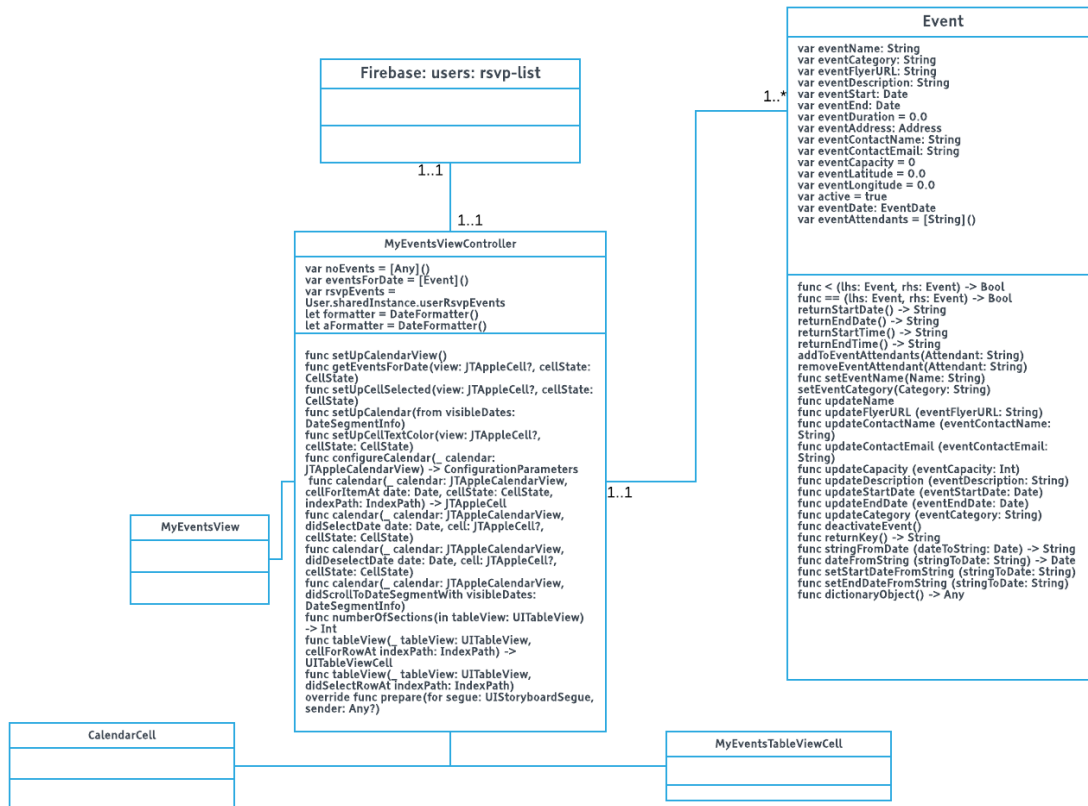
676 - Sign up**663 - Login****671 - Rsvp to Event**



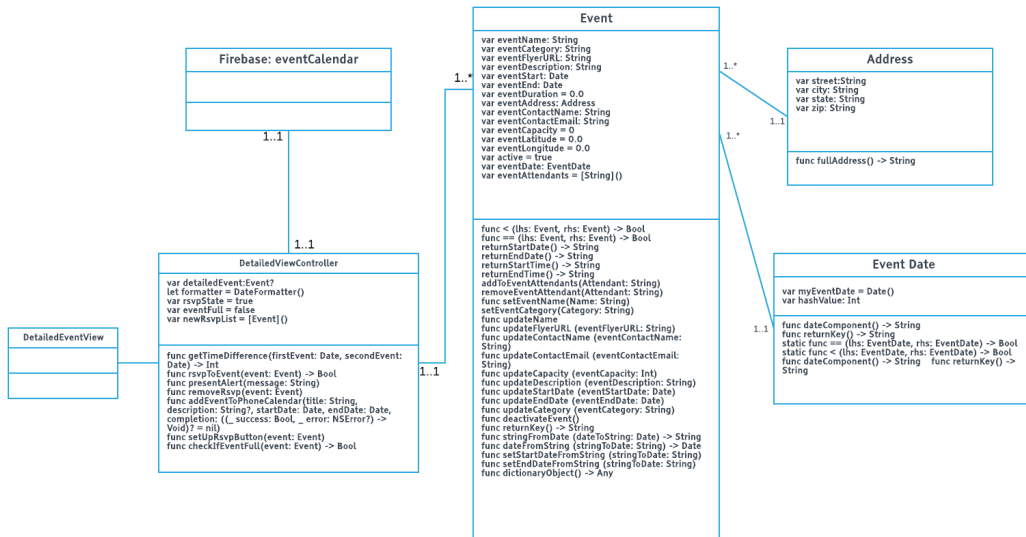
670 - View Event Details



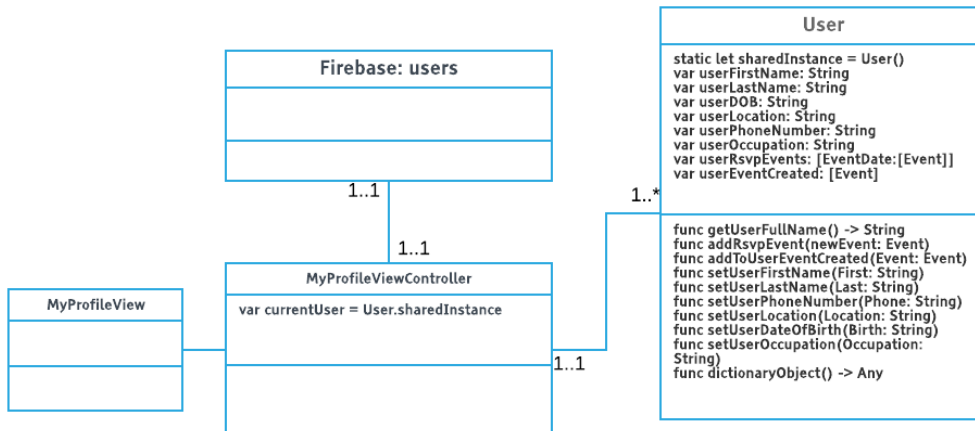
672 - View Rsvp Calendar



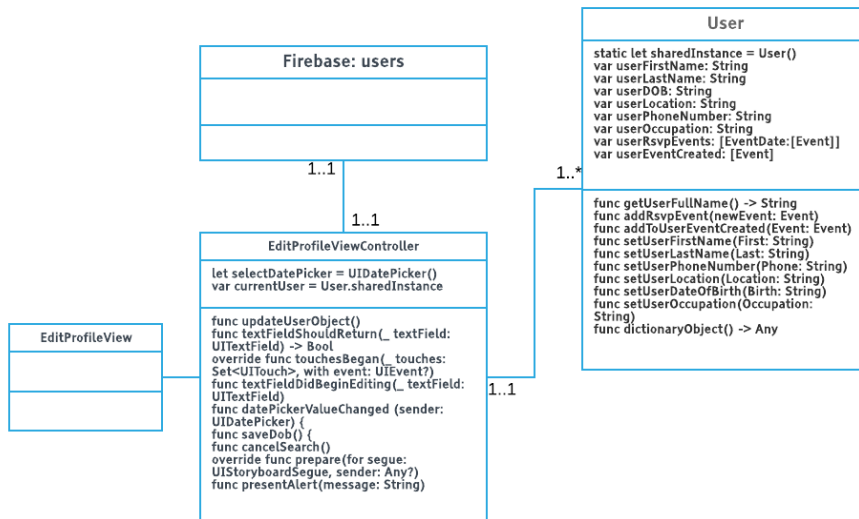
673 - Cancel Rsvp



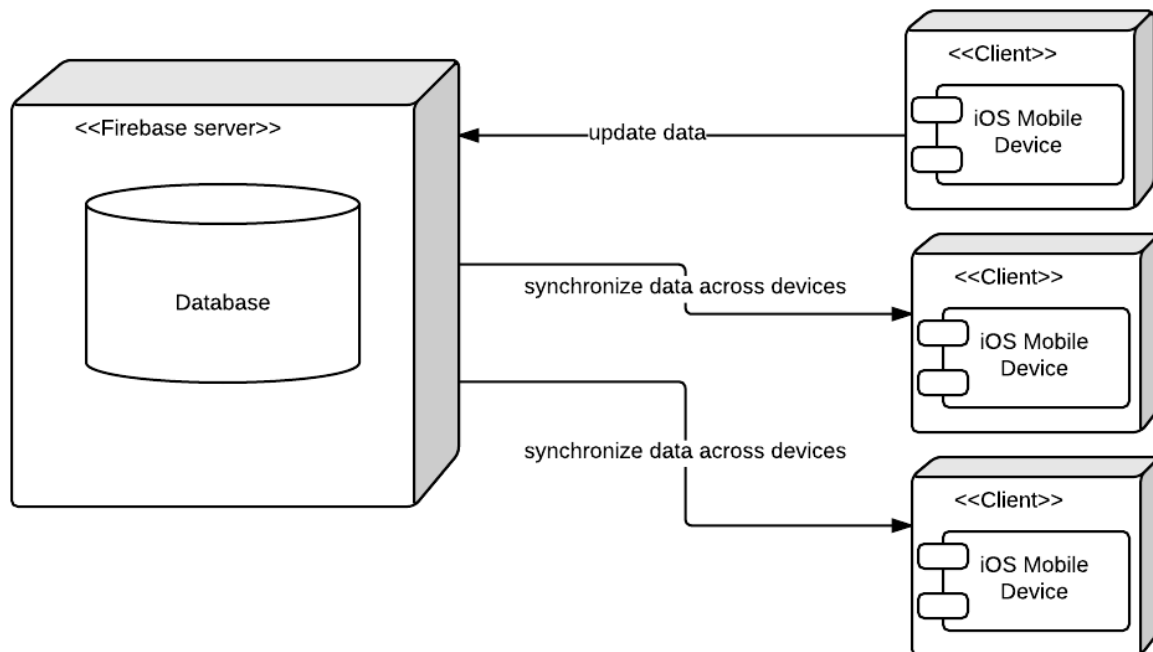
664 - View User Profile



665 - Update User Profile



Deployment Diagram



Design Patterns

During the implementation of the solution, the most common Cocoa design patterns were used in order to create the code that would be easy to understand and reuse.

We applied the principles of the Model View Controller design pattern, which is the building block of the iOS applications. Model View Controller pattern categorizes objects according to their role in the application, which allows separation of code based on these principles. In our code we put all of the model classes in a separate folder. All of the controller classes were grouped together based on their role as well. The majority of the view objects were designed in storyboard, which is “a visual representation of the user interface of an iOS application, showing screens of content and the connections between those screens”¹.

¹"Cocoa Application Competencies for iOS." *Storyboard*. N.p., 18 Sept. 2013. Web. 27 July 2017.

Another design pattern we utilized in the application is the Singleton pattern. When using Singleton class, an application always returns the same instance, regardless of how many times it was called. We implemented the User class and the EventCalendar class as Singletons because only one instance of each is required per process.

We used the Delegate pattern, which is an important one for iOS applications, as Apple uses it in most of the UIKit classes: UITableView, UITextField, UITextView and so on. Delegation allows one object, typically a Controller object, act on behalf of another one.

Since Firebase database in our application is a real time database, we used observer pattern to listen to the changes in the database.

System Validation

Unit Testing

Test case ID: TCEG01	Description/Summary of Test: Check View Event Calendar functionality, check that EventView loads with valid data from EventCalendar model class. Pre-condition: EventCalendar contains data. Expected Results: All events contained in EventCalendar are displayed in the table in chronological order, showing each event's name and date. Actual Result: As expected. Status (Fail/Pass): Pass.
Test case ID: TCEG02	Description/Summary of Test: Check View Map functionality, check that MapView loads with valid data from EventCalendar model class. Pre-condition: EventCalendar contains data. Expected Results: All events contained in EventCalendar are displayed correctly on the map, displaying a pin at each event's location. Actual Result: As expected. Status (Fail/Pass): Pass.
Test case ID: TCEG03	Description/Summary of Test: Check View Map functionality, check that user's current location is correctly displayed on the map. Pre-condition: Location services are enabled for the application.

	Expected Results: User's current location is displayed on the map. Actual Result: As expected. Status (Fail/Pass): Pass.
Test case ID: TCEG04.	Description/Summary of Test: Check Create Event functionality, check that the new event is created, all of the event properties are initialized, and new event is added to the EventCalendar. Pre-condition: EventCalendar does not contain event's information. Expected Results: Event is added to the EventCalendar with correct key, all of the properties are correctly initialized. Actual Result: As expected. Status (Fail/Pass): Pass.
Test case ID: TCEG05	Description/Summary of Test: Check Create Event functionality, check that when the new event is created, it is correctly displayed in the Events Calendar. Pre-condition: Events Calendar does not display event's information. Expected Results: New event's name, date, and time are correctly displayed in the Events Calendar table. Actual Result: As expected. Status (Fail/Pass): Pass.
Test case ID: TCEG06	Description/Summary of Test: Check Filter Map Results functionality, check that the date

	picker appears when the user clicks on the search bar. Pre-condition: The user has accessed the Map tab. Expected Results: When the user clicks on the search bar, the date picker is displayed at the bottom of the screen, showing the date picker for date only and two button on the toolabr on top of the date picker, “Cancel” and “Map”. Actual Result: As expected. Status (Fail/Pass): Pass.
Test case ID: TCEG07	Description/Summary of Test: Check Filter Map Results functionality, check that after the user clicks on the “Cancel” button on top of the date picker, the date picker view is dismissed, and the previous view of the map is presented to the user. Pre-condition: The user has accessed the Map tab and clicked on the search bar. Expected Results: When the user clicks on the “Cancel” button, the date picker is dismissed and previous view of the map is presented. Actual Result: As expected. Status (Fail/Pass): Pass.
Test case ID: TCEG08	Description/Summary of Test: Check Filter Map Results functionality, check that after the user selects the date for which to pam the results and clicks on the “Map” button on top of the date picker, only the events for the selected date are displayed, and the date is displayed in the search bar.

	Pre-condition: The user has accessed the Map tab, clicked on the search bar, and selected the date. Expected Results: When the user clicks on the “Map” button, the date picker is dismissed and only events on the selected date are pinned on the map; the search bar displays the date for which events are showed on the map. Actual Result: As expected. Status (Fail/Pass): Pass.
Test case ID: TCEG09	Description/Summary of Test: Check View Event Information on the Map functionality, check that once the user clicks on the pin on the map, the event’s information, such as name, date, and time, is shown. Pre-condition: The user has accessed the Map tab. Expected Results: Once the user clicks on the pin, the annotation for the pin displays the event’s information. Actual Result: As expected. Status (Fail/Pass): Pass.
Test case ID: TCEG10	Description/Summary of Test: Check View the List of the Events Created by the User functionality, check that EditEventsView loads with valid data from User’s userEventCreated property. Pre-condition: The user has clicked on the “Edit Events” button. Expected Results: All events contained in userEventsCreated array of the User class are

	displayed in the table in chronological order, showing each event's name and date. Actual Result: As expected. Status (Fail/Pass): Pass.
Test case ID: TCEG11	Description/Summary of Test: Check Edit Event Created by the User functionality, check that the event being edited updates the changed properties. Pre-condition: The user has selected the event to edit from the list of the events created by the user. Expected Results: Event properties are updated. Actual Result: As expected. Status (Fail/Pass): Pass.
Test case ID: TCKB01	Description/Summary of Test: Check that event details view loads with valid data from event model class. Precondition: An event exists. Expected Results: All event information is displayed correctly in the view. Actual Result: As expected. Status (Fail/Pass): Pass.
Test Case ID: TCKB02	Description/Summary of Test: Check that the flyer URL and email address in event details view opens the appropriate application on iPhone. Precondition: An event exists. Expected Results: URLs open the corresponding application.

	Actual Result: As expected. Status (Fail/Pass): Pass.
Test Case ID: TCKB03	Description/Summary of Test: Check that the event is added to the user rsvp events dictionary in the user model class. Precondition: An event exists. Expected Results: The event is added to the user rsvp dictionary. Actual Result: As expected. Status (Fail/Pass): Pass.
Test Case ID: TCKB04	Description/Summary of Test: Check that the user is added to the attendees array in the event model class. Precondition: Rsvp button was clicked once. Expected Results: The user is added to the attendees array. Actual Result: As expected. Status (Fail/Pass): Pass.
Test Case ID: TCKB05	Description/Summary of Test: Check that the event details view displays a cancel button instead of an rsvp button and that the user appears in the attendees list. Precondition: Rsvp button was clicked once. Expected Results: Cancel button is displayed correctly and user name appears in list. Actual Result: As expected. Status (Fail/Pass): Pass.
Test Case ID: TCKB06	Description/Summary of Test: Check that clicking a calendar cell loads events in the table view below calendar.

	Precondition: Event has been RSVP'd to. Expected Results: Events appear in the list for the selected calendar cell. Actual Result: As expected. Status (Fail/Pass): Pass.
Test Case ID: TCKB07	Description/Summary of Test: Check that clicking cancel in the event details view removes the user's name from the attendees list in event details. Precondition: Event has been RSVP'd to. Expected Results: User was successfully removed from attendees list in event details. Actual Result: As expected. Status (Fail/Pass): Pass.
Test Case ID: TCKB08	Description/Summary of Test: Check that user information is displayed correctly in the My Profile tab. Precondition: User must have registered and provided user information. Expected Results: All of the user information from the database is displayed correctly in the My profile tab. Actual Result: As expected. Status (Fail/Pass): Pass.
Test Case ID: TCKB09	Description/Summary of Test: Check that if the user enters incorrect data in the sign up form, the user receives validation and is blocked from registering an account. Precondition: None. Expected Results: User should be blocked from registering and appropriate error

	messages are displayed. Actual Result: As expected. Status (Fail/Pass): Pass.
Test Case ID: TCKB10	Description/Summary of Test: Check that validation for required fields such as firstname and lastname are provided when user edits profile. Check that other fields are not required but when filled in also validate the user input. Precondition: User is registered and signed in. Expected Results: The edit profile view has the appropriate validation, blocks user from updating profile and displays clear error messages. Actual Result: As expected. Status (Fail/Pass): Pass.

In addition to unit tests, a series of manual tests were performed to verify that once the user is logged in to the application and the connection to the Firebase database is established, the application successfully retrieves data from the database and loads it into the application model to display event calendar, event details, and user details.

GLOSSARY

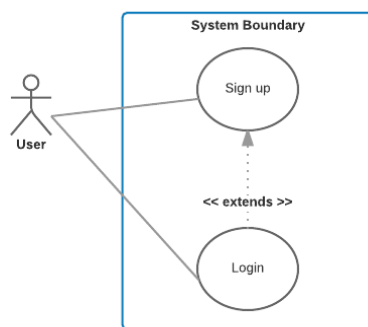
- Actors: any end-user or external entity that interacts with the system.
- System architecture: a conceptual model that defines the structure, behavior, and more views of a system.
- Deployment Diagram: a structure diagram which shows architecture of the system as deployment (distribution) of software artifacts to deployment targets.
- Interface: a device or program enabling a user to communicate with a computer.
- Repository: a central location in which data is stored and managed.
- Sequence Diagram: sequence of interactions among objects in use case.
- Software testing: the process of operating software under specified conditions, observing or recording the results and making an evaluation of some aspect of the software.
- Storage: the action or method of storing something for future use.
- System: a set of connected things or parts forming a complex whole.
- Task: an atomic unit of work that can be managed and consumes resources.
- User Story: a description of an objective a person should be able to achieve, or a feature that a person should be able to utilize, when using a software application.
- Acceptance Criteria: specific user tasks, functions or business processes that must be in place.
- Use Case Diagram: diagram that shows actors, use cases, and all their possible relationships.

APPENDIX

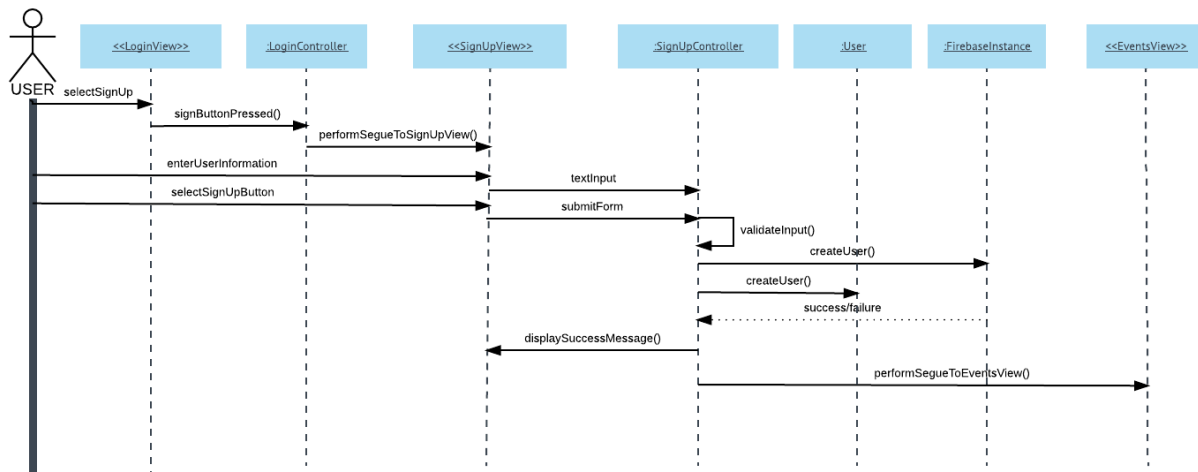
Appendix A - UML Diagrams

676 - Sign up

Use case diagram:



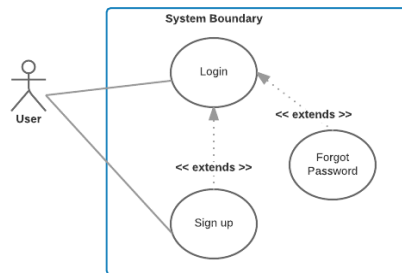
Sequence diagram:



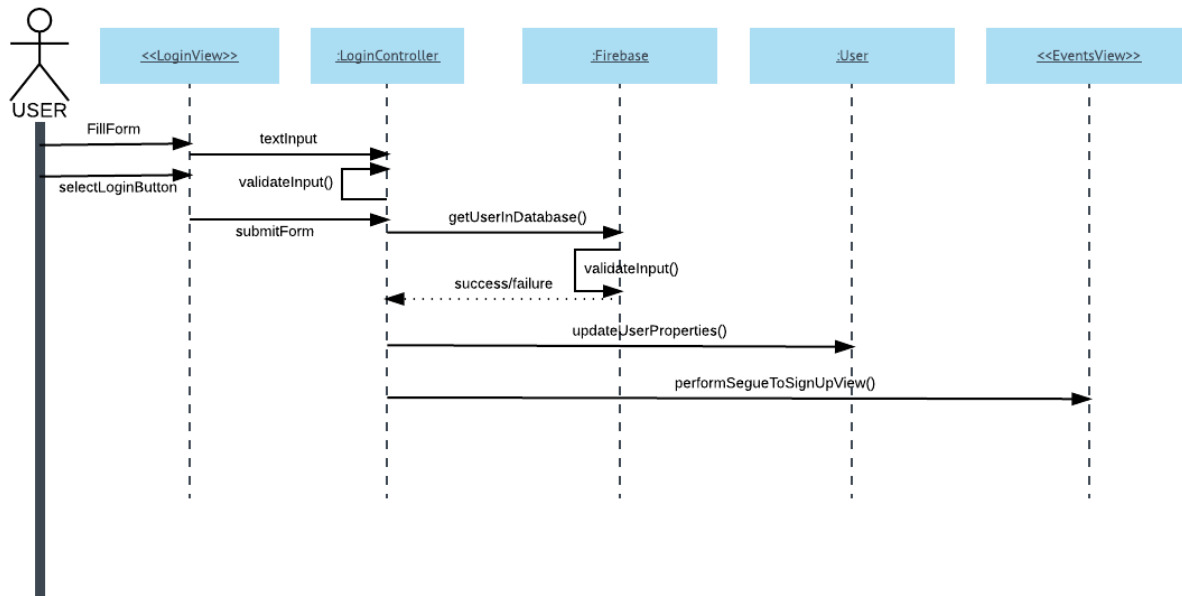
663 - Login

Use case diagram:

Date

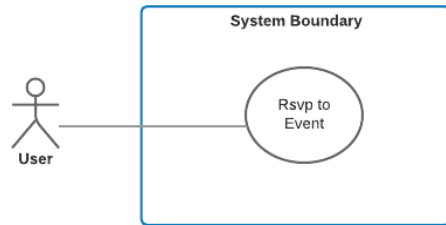


Sequence diagram:

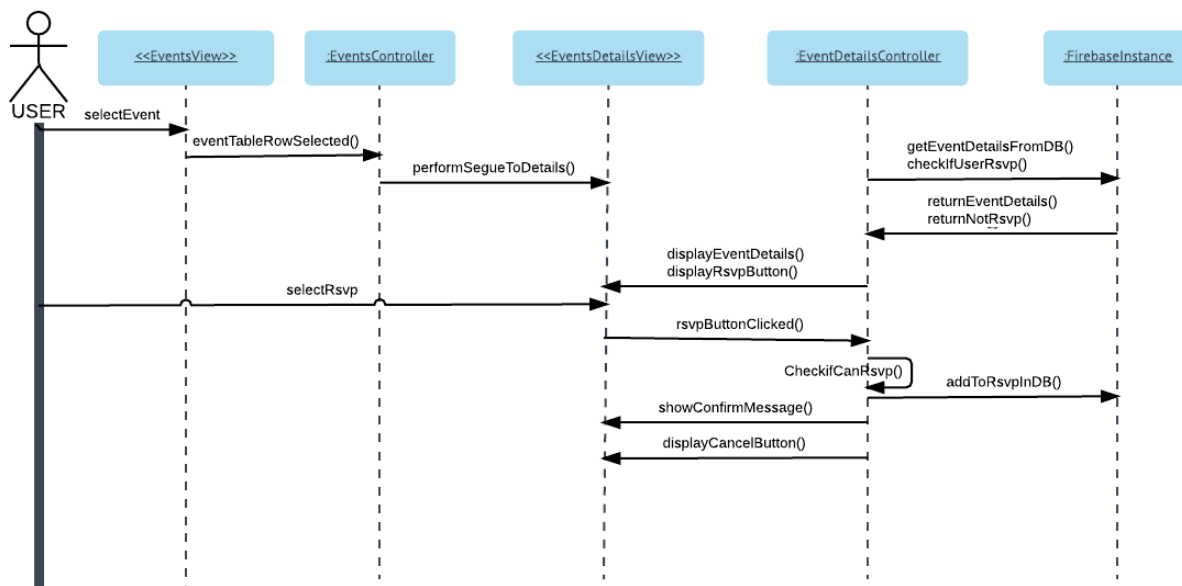


671 - Rsvp to Event

Use case diagram:

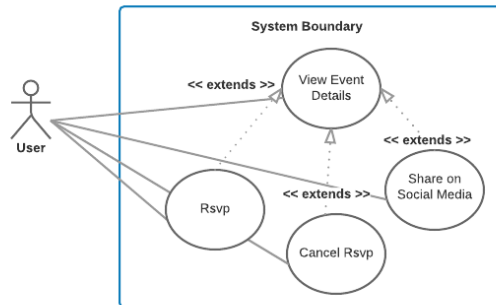


Sequence diagram:

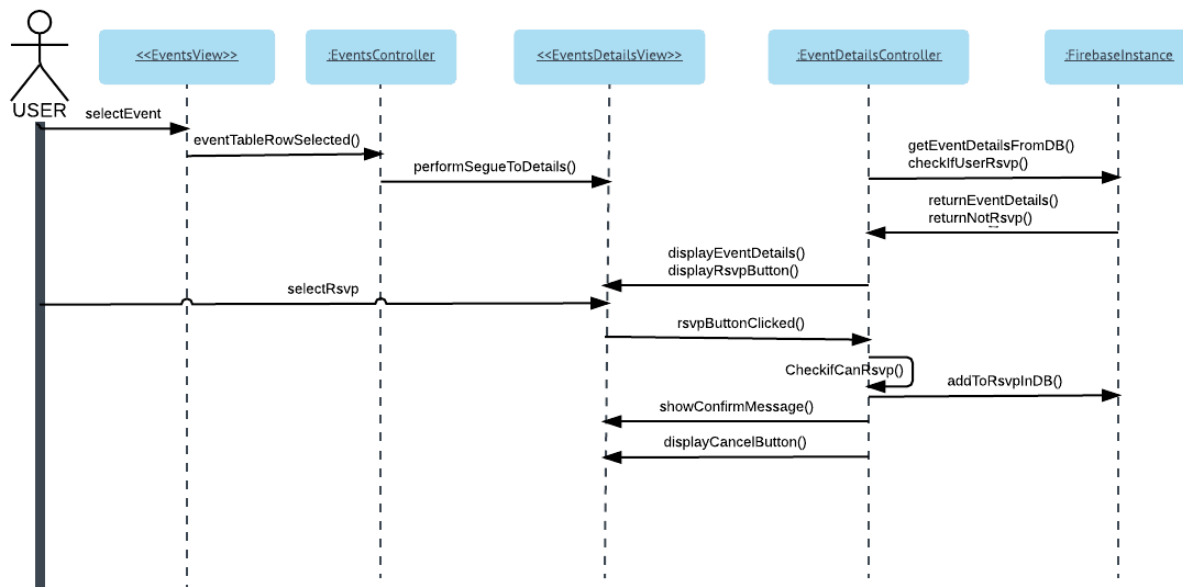


670 - View Event Details

Use case diagram:

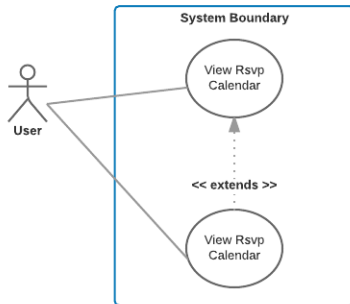


Sequence diagram:

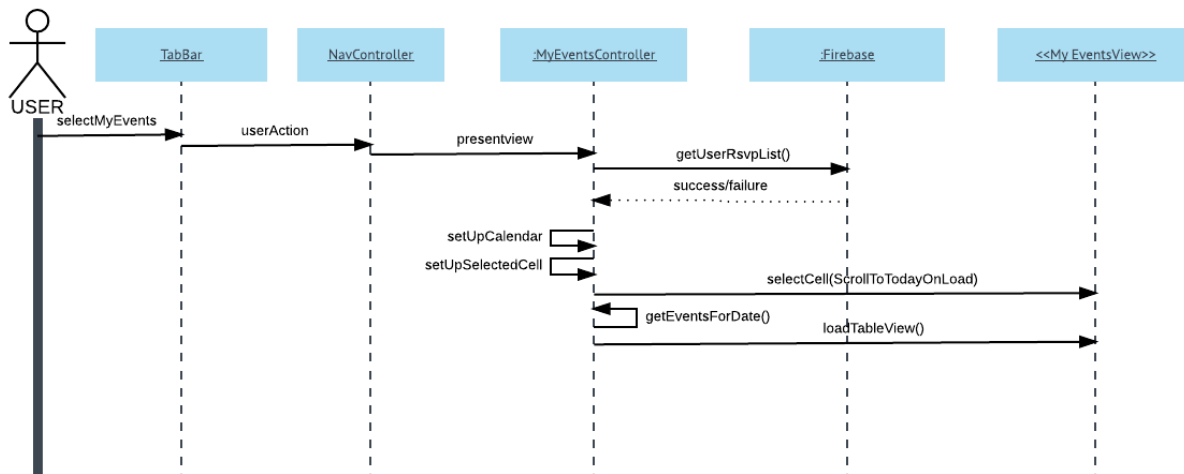


672 - View Rsvp Calendar

Use case diagram:

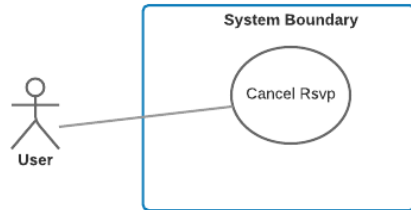


Sequence diagram:

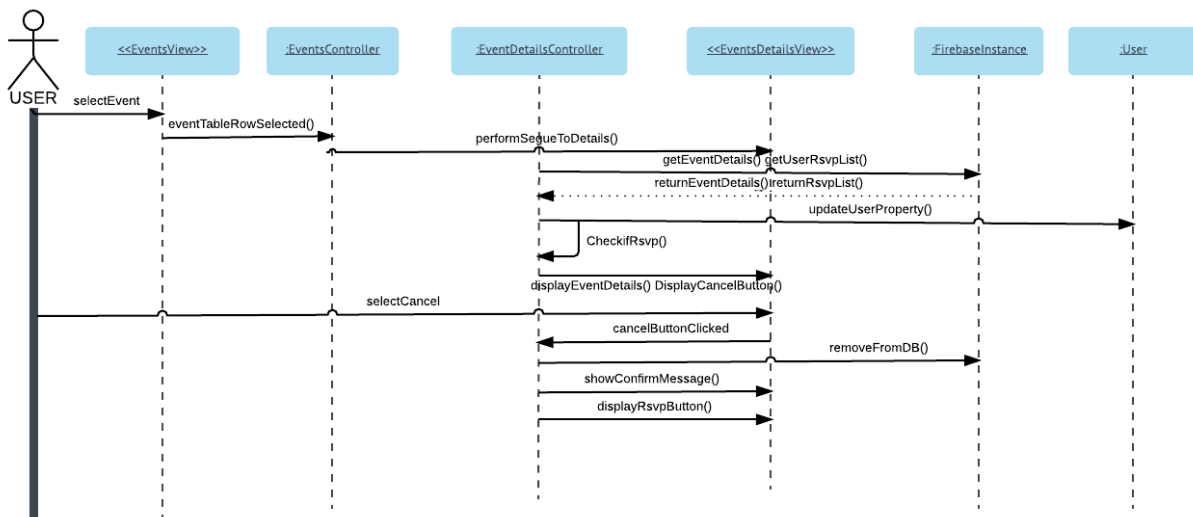


673 - Cancel Rsvp

Use case diagram:

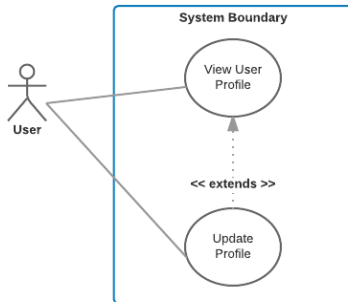


Sequence diagram:

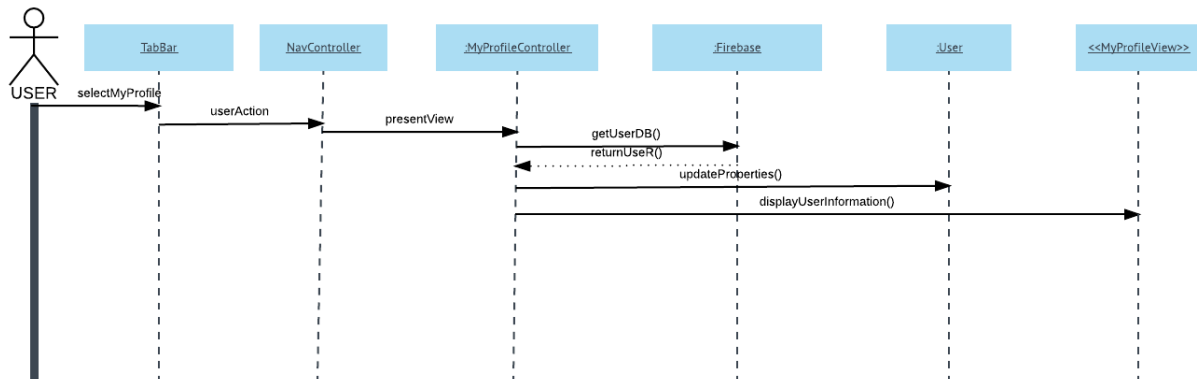


664 - View User Profile

Use case diagram:

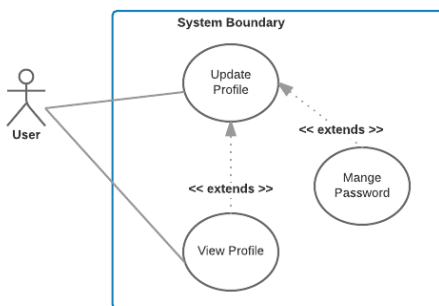


Sequence diagram:

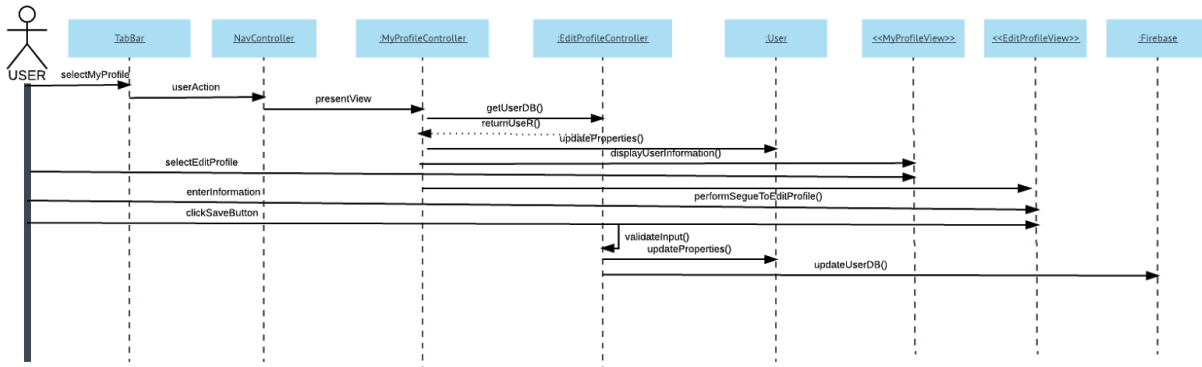


665 - Update User Profile

Use case diagram:

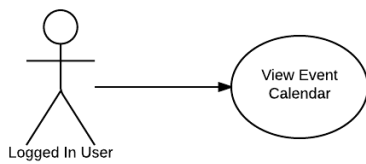


Sequence diagram:

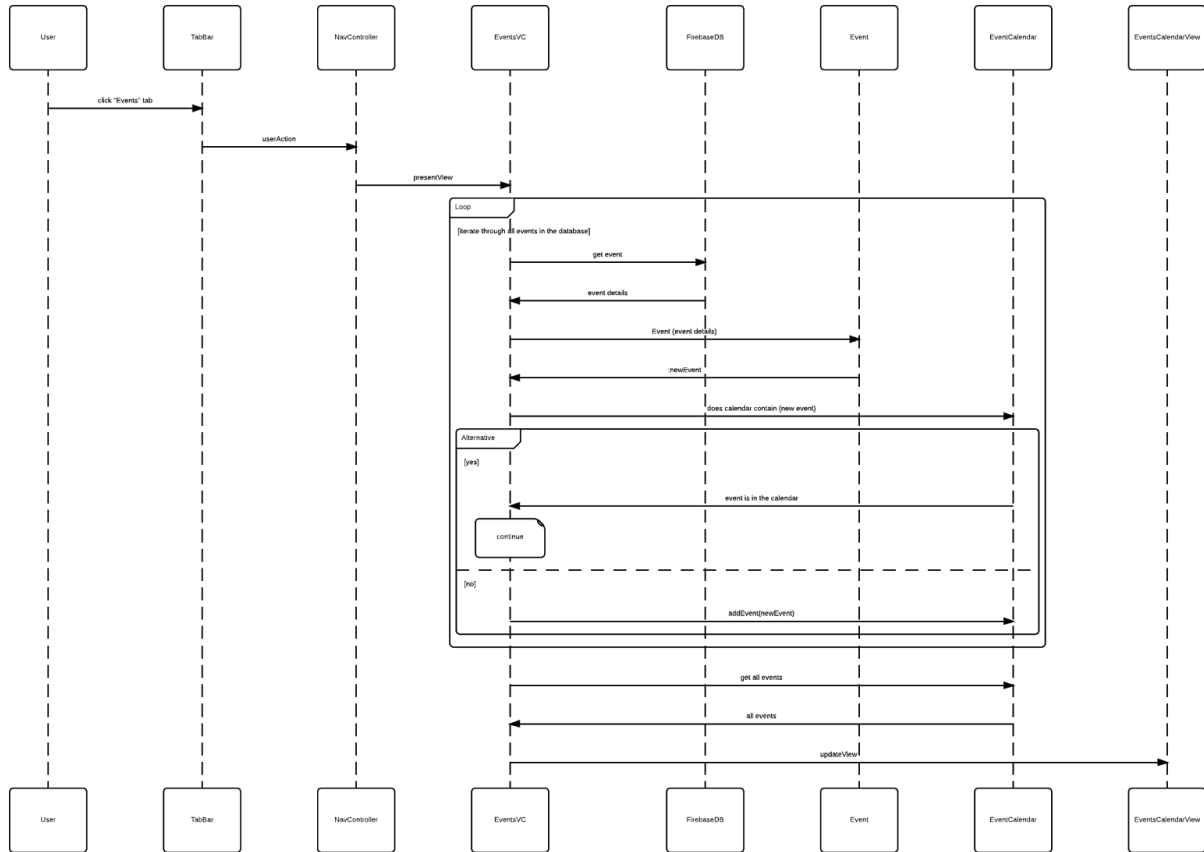


668 - View Event Calendar

Use case diagram:

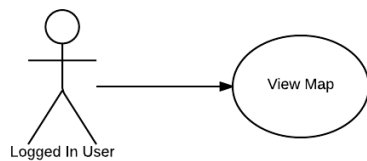


Sequence diagram:

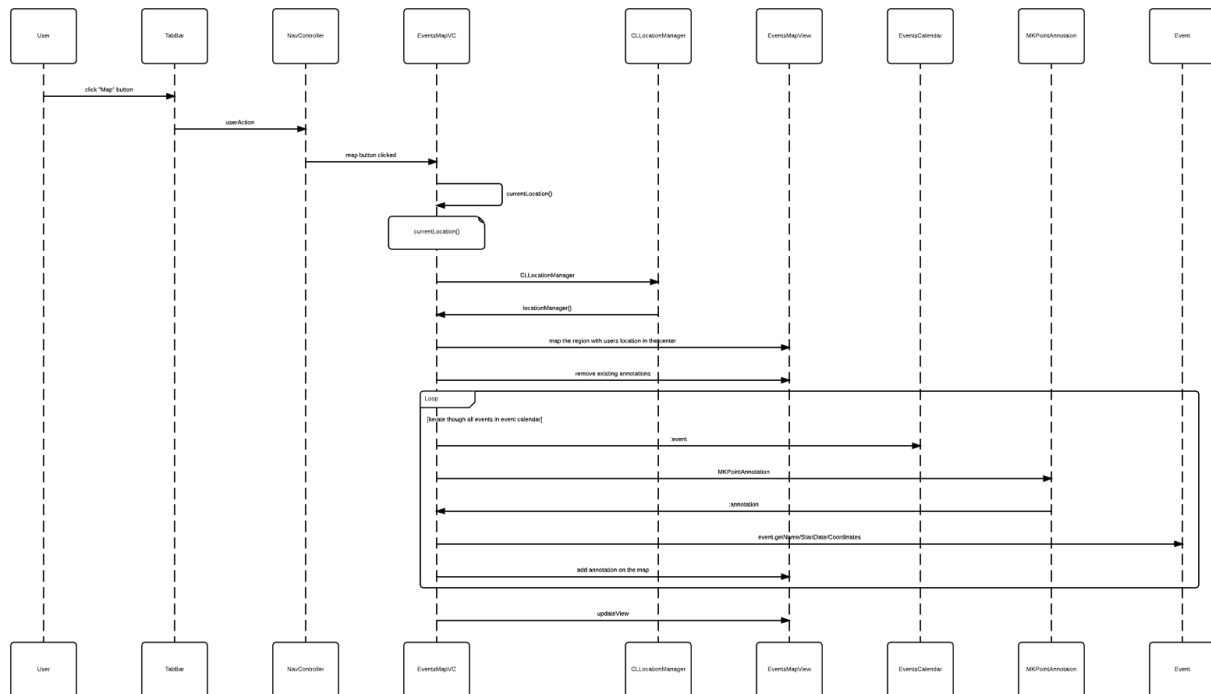


677 - View Map

Use case diagram:

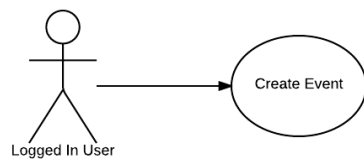


Sequence diagram:

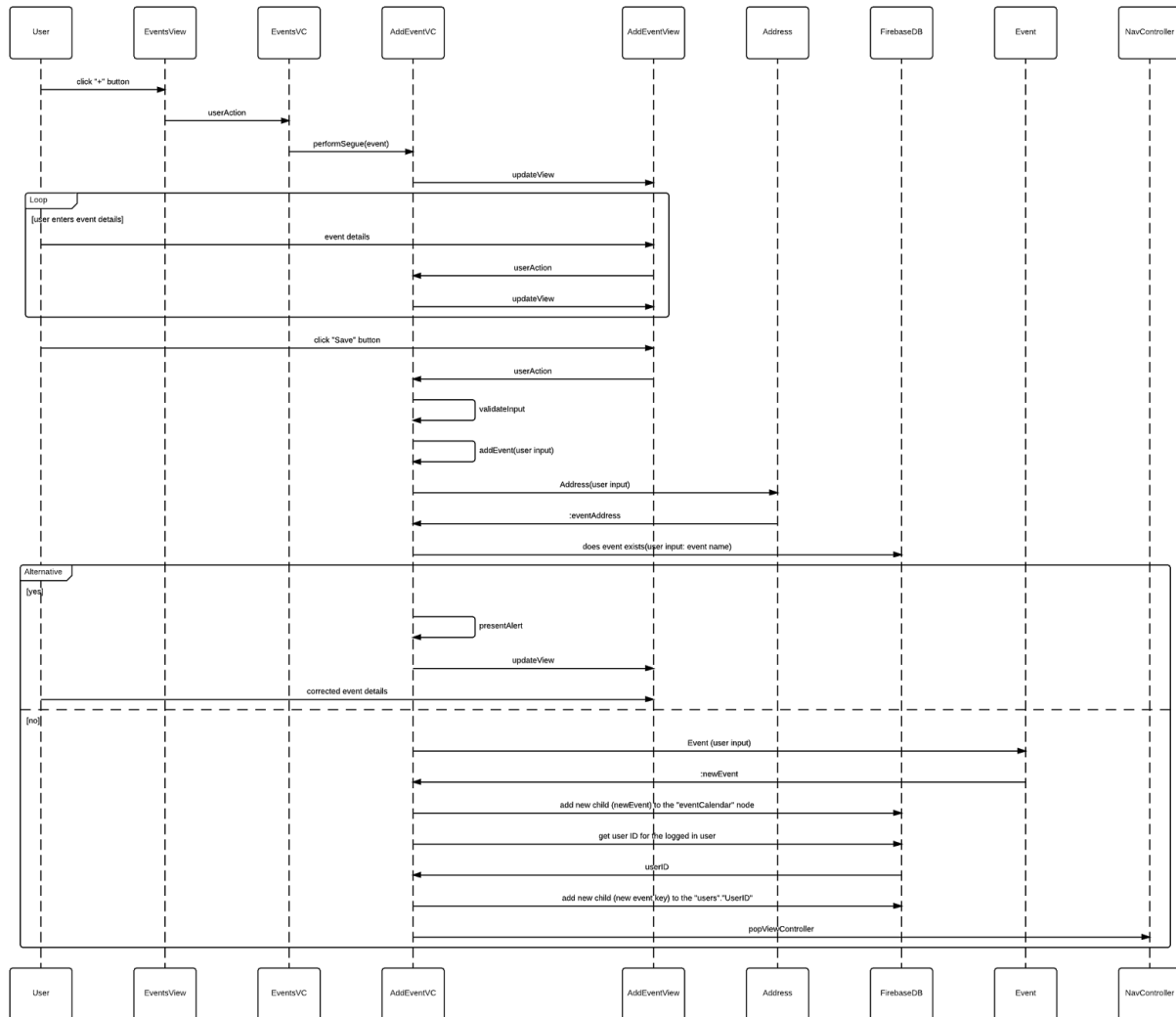


667 - Create Event

Use case diagram:

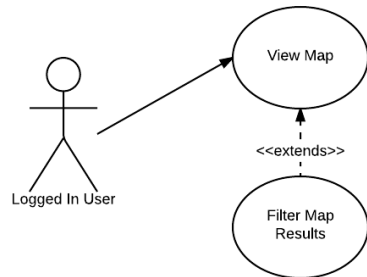


Sequence diagram:

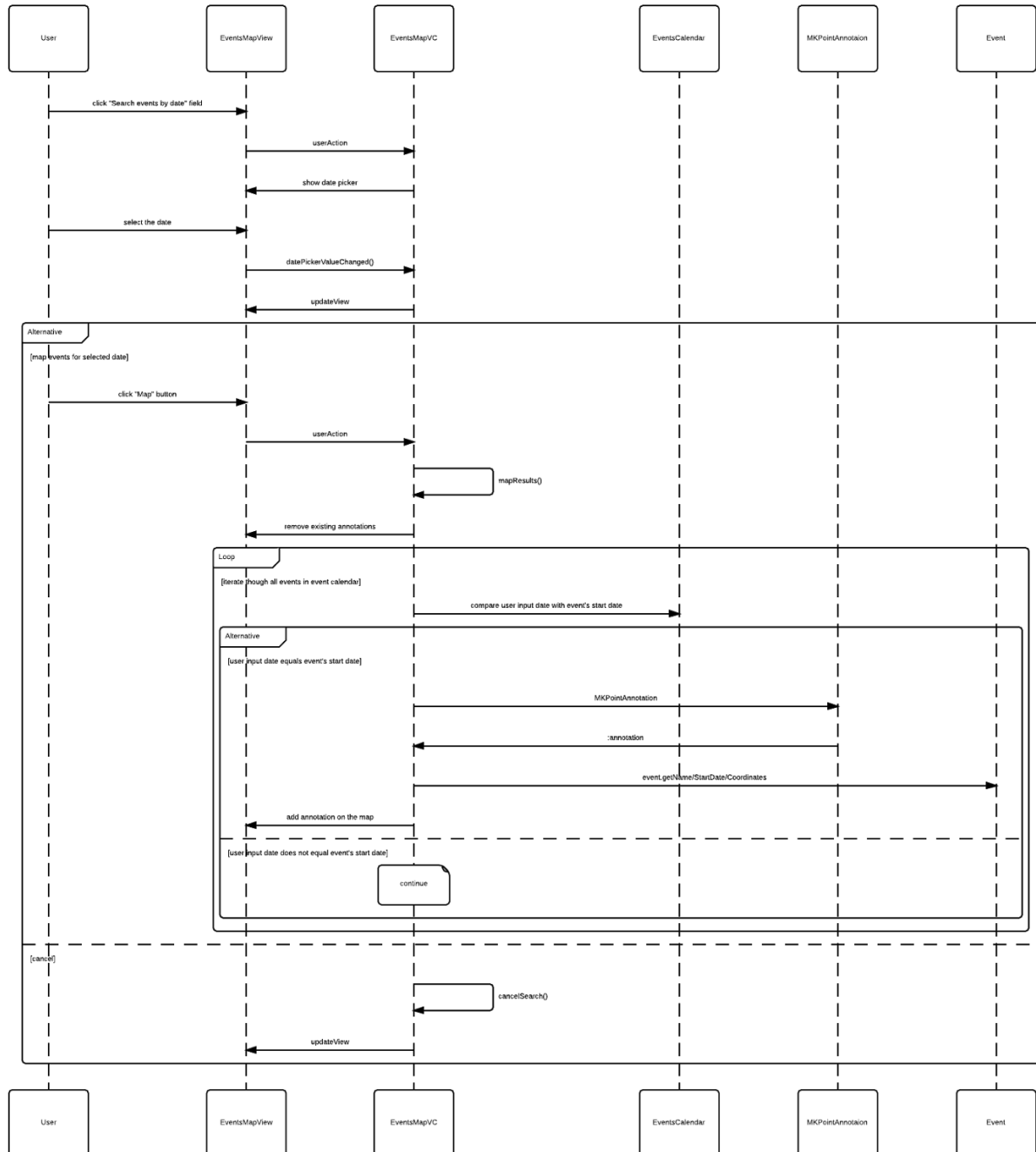


674 - Filter Map Results

Use case diagram:

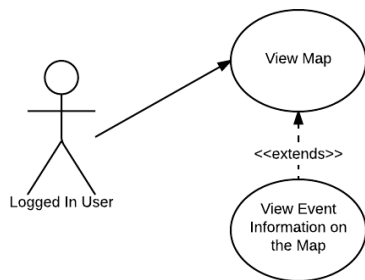


Sequence diagram:

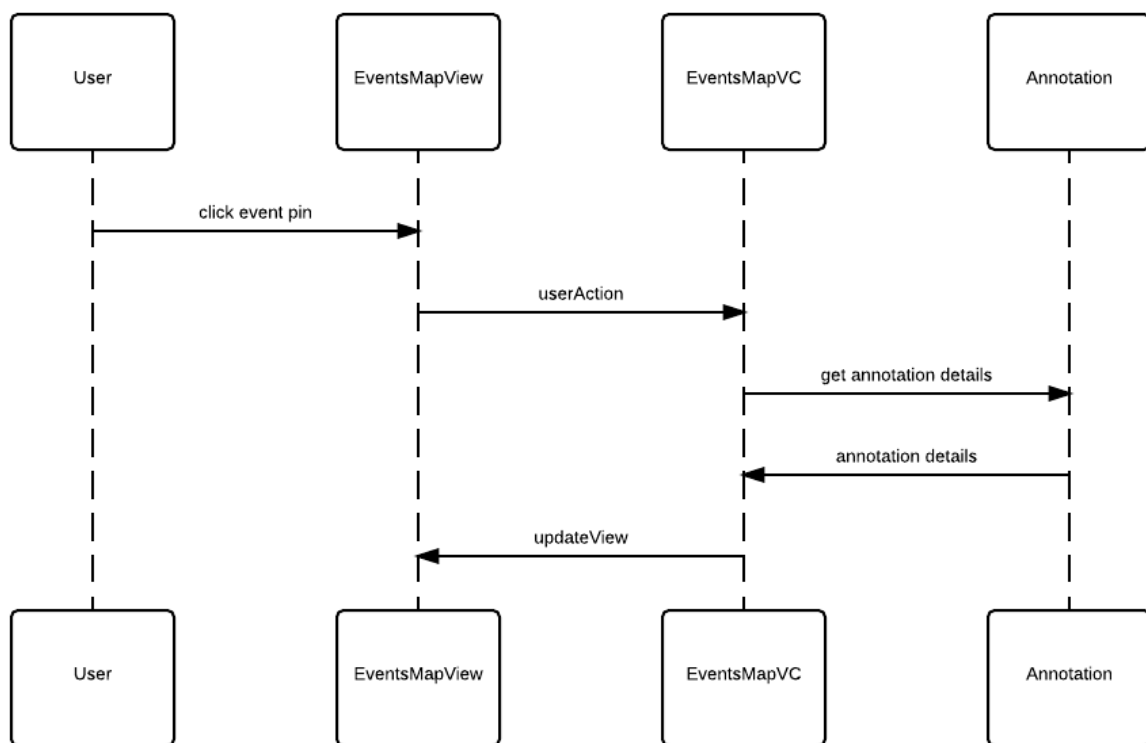


673 - View Event Information on the Map

Use case diagram:

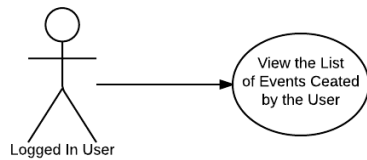


Sequence diagram:

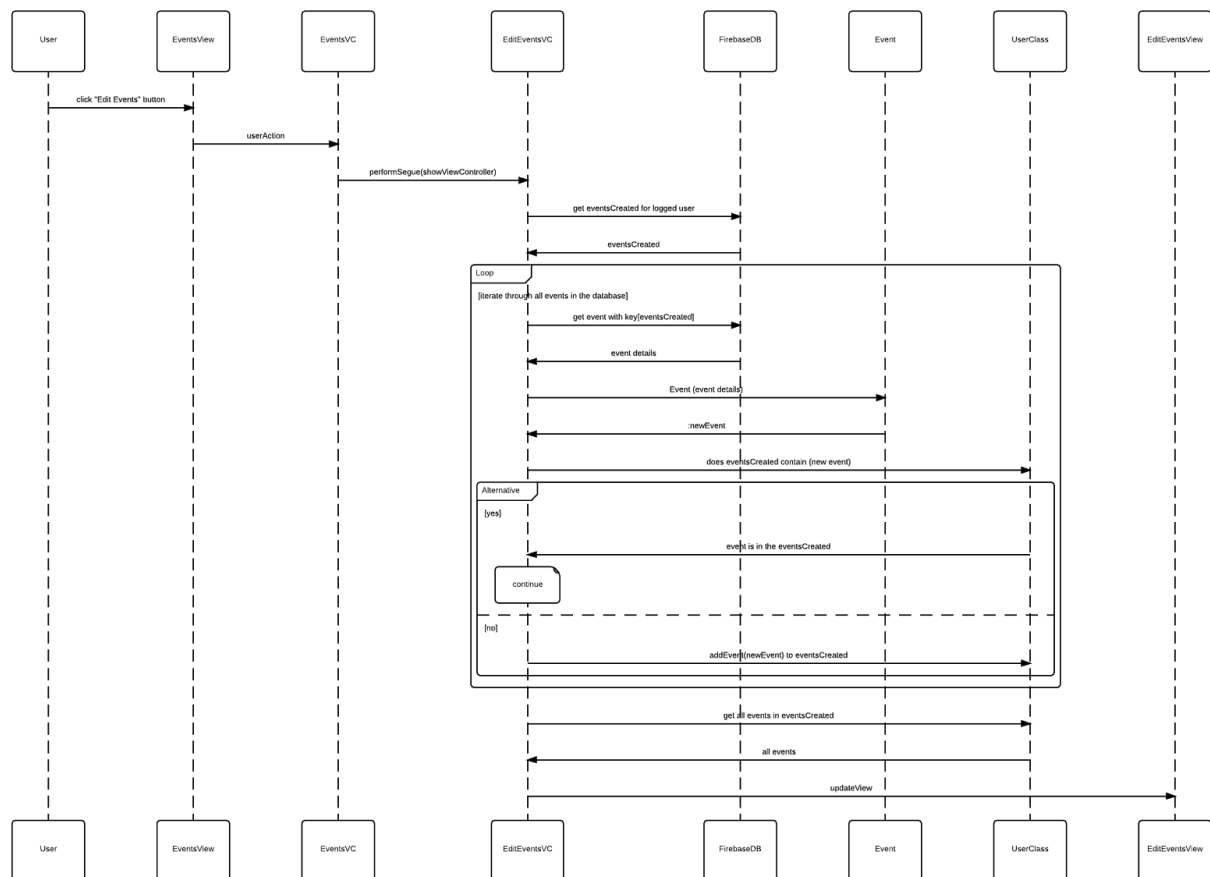


679 - View the List of the Events Created by the User

Use case diagram:

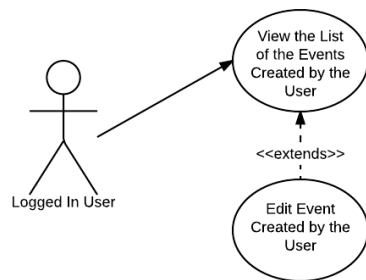


Sequence diagram:

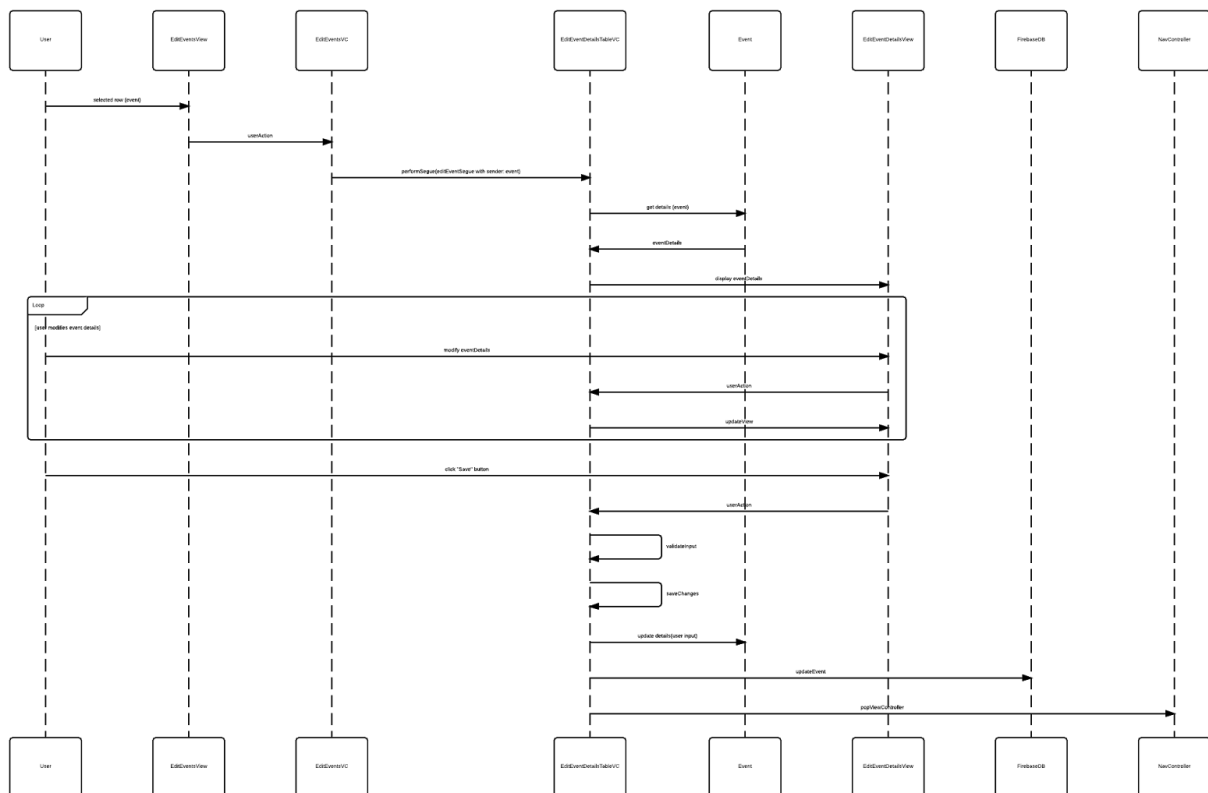


678 - Edit Event Created by the User

Use case diagram:



Sequence diagram:



Appendix B - User Interface Design

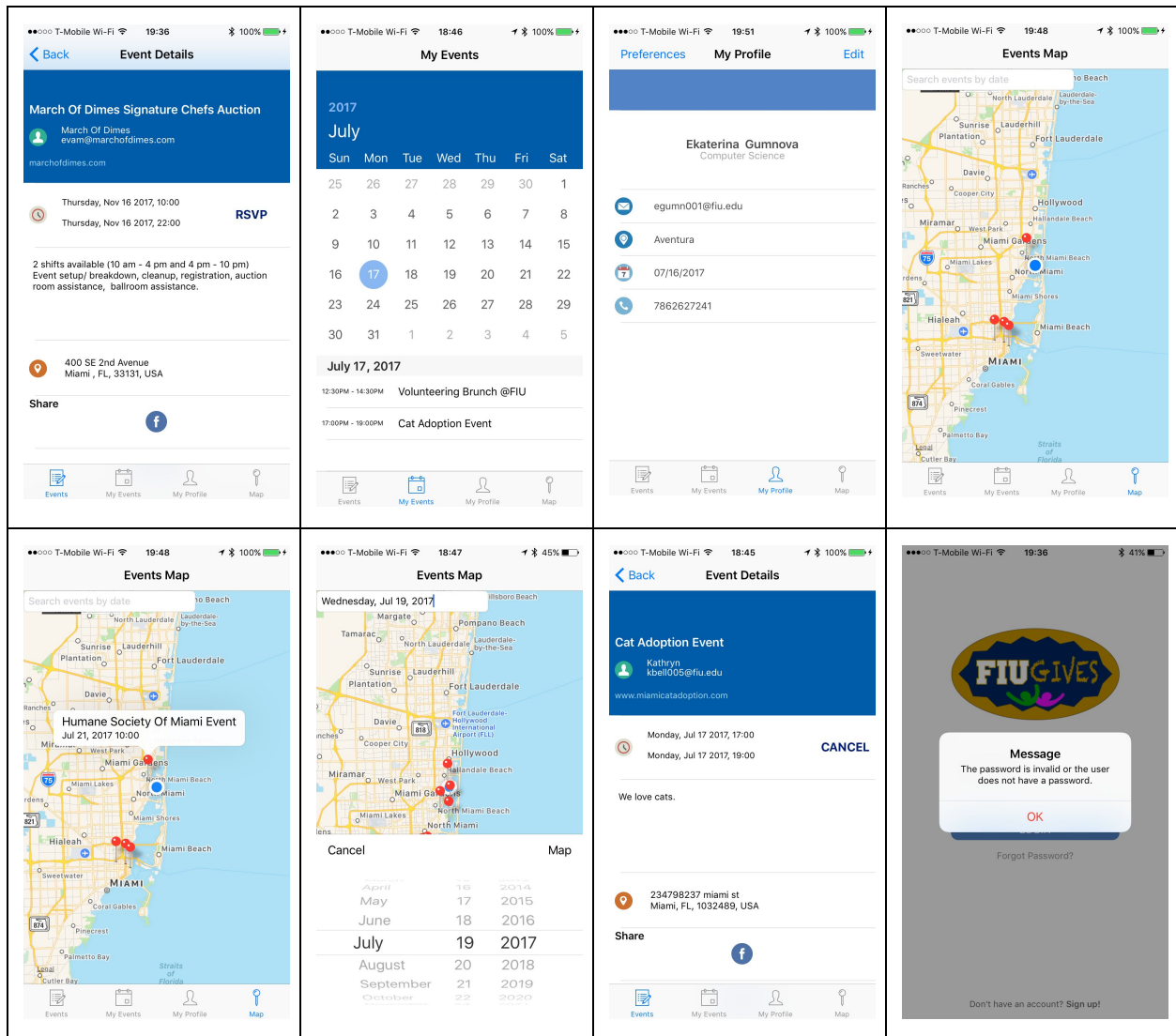
The image displays eight mobile app interface mockups for the FIU GIVES application, arranged in a 2x4 grid. Each mockup shows a different screen of the app, including login, sign-up, event calendar, event details, and event creation. The app features a blue and yellow color scheme with the FIU GIVES logo. The top status bar shows 'T-Mobile Wi-Fi', '00:48', and '100%' battery. The bottom navigation bar includes icons for 'Events', 'My Events', 'My Profile', and 'Map'.

Top Row Mockups:

- Sign Up Screen:** Features the FIU GIVES logo and a form with fields for Email, Password, Confirm Password, Firstname, Lastname, Date of Birth, Phone Number, Location, and Occupation. A 'SIGN UP' button is at the bottom.
- Login Screen:** Features the FIU GIVES logo and a form with fields for Email and Password. A 'LOGIN' button and a 'Forgot Password?' link are at the bottom.
- Events Calendar:** Displays a calendar view for July 21-23, 2017, with event listings for each day.
- Edit Events:** Displays a list of events with details like 'Volunteering Orientation @ Biscayne Bay...', 'Petsmart Aventura Adoption Event', and 'Art Walk In Design District'.

Bottom Row Mockups:

- Edit Events:** A form for editing an event, including fields for Event Name, Event Flyer URL, Event Contact Name, Event Contact Email, Street Address, City, State, Zip, Set Event Capacity, and Description.
- Add Event:** A form for adding a new event, including fields for Event Name, Event Flyer URL, Event Contact Name, Event Contact Email, Street Address, City, State, Zip, Set Event Capacity, and Description.
- Add Event:** A form for adding a new event, including fields for Event Name, Event Flyer URL, Event Contact Name, Event Contact Email, Street Address, City, State, Zip, Set Event Capacity, and Description.
- Add Event:** A form for adding a new event, including fields for Event Name, Event Flyer URL, Event Contact Name, Event Contact Email, Street Address, City, State, Zip, Set Event Capacity, and Description.



Appendix C - Sprint Review Reports

Sprint Review Meeting Minutes: 05/26/2017

Attendees: Kathryn Bello, Ekaterina Gumnova
Start time: 7:00 PM
End time: 10:00 PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

None were accepted -- have not met with project managers.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

None were rejected -- have not met with project managers.

Sprint Review Meeting Minutes: 06/09/2017

Attendees: Kathryn Bello, Ekaterina Gumnova
Start time: 5:00 PM
End time: 8:00 PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

None were accepted -- have not met with project managers.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

None were rejected -- have not met with project managers.

Sprint Review Meeting Minutes: 06/23/2017

Attendees: Kathryn Bello, Ekaterina Gumnova

Start time: 8:00 PM

End time: 9:00 PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners: All.

View Event Details #670

RSVP Event #671

View RSVP Calendar #672

View Map #677

View Event List #668

Create Event #667

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

None were rejected

Sprint Review Meeting Minutes: 07/07/2017

Attendees: Kathryn Bello, Ekaterina Gumnova

Start time: 9:00 PM

End time: 10:00 PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners: All.

Filter Map Results #674

View Profile #664

Set Preferences in Profile #665

Update Profile #666

Login #663

Signup #676

View the List of the Events Created by the User #679

Edit Event Created by the User #678

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

None were rejected.

Appendix D - User Manuals, Installation/Maintenance Document, Shortcomings/Wishlist Document and other documents

Shortcomings

User story #664 - View User Profile	When the user clicks on the User tab, the user's information is not fully displayed, although all of it is updated in the database.
User story #678 - Edit Event Created by the User	When the user clicks "Save" button after finishing editing the event, the changes are not saved to the database. Function saveChanges() is implemented to only save changes locally in the model.

Wishlist

User story #675 - Share Event on Social Media	As a user, I would like to share an event on social media platforms so that I can promote the volunteer opportunity with all of my friends/followers.
User story #665 - Set Preferences in Profile	As a user, I would like to set my preferences for what event categories I am interested in my profile so that I can only see events according to my set preferences.

REFERENCES

1. "Cocoa Application Competencies for iOS." *Storyboard*. N.p., 18 Sept. 2013. Web. 27 July 2017.
<<https://developer.apple.com/library/content/documentation/General/Conceptual/DevPedia-CocoaApp/Storyboard.html>>.
2. "Cocoa Core Competencies." *Singleton*. N.p., 21 Oct. 2015. Web. 27 July 2017.
<<https://developer.apple.com/library/content/documentation/General/Conceptual/DevPedia-CocoaCore/Singleton.html>>.
3. "Cocoa Core Competencies." *Delegation*. N.p., 21 Oct. 2015. Web. 27 July 2017.
<<https://developer.apple.com/library/content/documentation/General/Conceptual/DevPedia-CocoaCore/Delegation.html>>.
4. *Apple Developer Documentation*. N.p., n.d. Web. 28 July 2017.
<<https://developer.apple.com/documentation/>>.
5. "Featured Tutorials and Announcements." *Ray Wenderlich | Tutorials for iPhone / iOS Developers and Gamers*. N.p., n.d. Web. 28 July 2017.
<<https://www.raywenderlich.com/>>.
6. "Documentation | Firebase." *Google*. Google, n.d. Web. 28 July 2017.
<<https://firebase.google.com/docs/>>.