

|  |
| --- |
| SRS Project |
|  |
| November 9  Grad Student Team  Members: Tom Grimes, Hein Tran, Ram Tamanampudi, Likitha Varapana |

SWEG 5301

Contents

[1. Executive Summary 5](#_Toc87381538)

[Summary 5](#_Toc87381539)

[2. System Request 6](#_Toc87381540)

[System Request – Application: Fairfield Connect 6](#_Toc87381541)

[3. Feasibility Analysis 7](#_Toc87381542)

[Technical Feasibility 7](#_Toc87381543)

[Economic Feasibility 7](#_Toc87381544)

[Operational Feasibility 8](#_Toc87381545)

[4.Requirements Definition 9](#_Toc87381546)

[Functional Requirements 9](#_Toc87381547)

[Login 9](#_Toc87381548)

[Forgot Password 9](#_Toc87381549)

[Create New Account 10](#_Toc87381550)

[Navigation 10](#_Toc87381551)

[Create post 11](#_Toc87381552)

[View/Sort Post 11](#_Toc87381553)

[Add Comment 11](#_Toc87381554)

[Non-Functional Requirements 12](#_Toc87381555)

[Ease of use 12](#_Toc87381556)

[Notifications 12](#_Toc87381557)

[Security 12](#_Toc87381558)

[Response Time 12](#_Toc87381559)

[Portability 13](#_Toc87381560)

[5. Model Driven Architecture 14](#_Toc87381561)

[Model Driven Architecture 14](#_Toc87381562)

[6. Functional Model 15](#_Toc87381563)

[Use Case Descriptions 15](#_Toc87381564)

[7. Structural Model 28](#_Toc87381565)

[CRC Cards 28](#_Toc87381566)

[Student 28](#_Toc87381567)

[User 29](#_Toc87381568)

[Post 30](#_Toc87381569)

[Thumbnail 31](#_Toc87381570)

[Notification 32](#_Toc87381571)

[Section 33](#_Toc87381572)

[Category 34](#_Toc87381573)

[Landlord 35](#_Toc87381574)

[Review 36](#_Toc87381575)

[Landlord Request 37](#_Toc87381576)

[Class Diagrams 38](#_Toc87381577)

[8. Behavioral Model 41](#_Toc87381578)

[Sequence Diagrams 41](#_Toc87381579)

[State Diagrams 51](#_Toc87381580)

[Create User Account: 51](#_Toc87381581)

[User Sign In : 51](#_Toc87381582)

[Edit or Delete: 51](#_Toc87381583)

[Review Land Lord: 52](#_Toc87381584)

[Create Admin Account: 52](#_Toc87381585)

[9. Prototype 54](#_Toc87381586)

[Prototype Screenshots 54](#_Toc87381587)

[Public Home Page 54](#_Toc87381588)

[Sign In Page 54](#_Toc87381589)

[Sign In (Failed) 55](#_Toc87381590)

[Create Account 55](#_Toc87381591)

[Successful Account Creation 56](#_Toc87381592)

[Account Verified 56](#_Toc87381593)

[Create Account Errors 57](#_Toc87381594)

[Forgot Password 57](#_Toc87381595)

[Set New Password 58](#_Toc87381596)

[Confirm New Password 58](#_Toc87381597)

[Internal Home Page 59](#_Toc87381598)

[Admin Home Page 59](#_Toc87381599)

[Community Page 60](#_Toc87381600)

[Tutoring Page 60](#_Toc87381601)

[Marketplace Page 61](#_Toc87381602)

[Housing Page 61](#_Toc87381603)

[Landlords Page 62](#_Toc87381604)

[Sorting Landlords 62](#_Toc87381605)

[Housing Reviews 63](#_Toc87381606)

[Sorting Housing Reviews 63](#_Toc87381607)

# 1. Executive Summary

|  |
| --- |
| Summary Text about the summary |

# 2. System Request

## System Request – Application: Fairfield Connect

|  |  |
| --- | --- |
| Project sponsor: | Grad Student Team |
| Business need: | Currently, no application is available to students for connecting with other verified students for all of the things needed to live a successful student life. There are currently options for finding housing, tutors, or someone to sell your old TV to, but none that are free of sponsored astroturfing and spam. |
| Business Requirements: | This application will allow only verified students to create a user account. With this account they will be able to make and look at posts for connecting with a roommate, leave and read reviews of landlords and housing facilities, make and view posts for connecting with other students for tutoring or simply socializing, and list items for sale. Users can leave comments directly on the posts or request a direct message with the poster. |
| Business Value: | When implemented, this application will improve the quality of life and strengthen the community of the Fairfield student body. Being able to get honest feedback not polished by greedy landlords and instead from fellow students will lead to users being happier with their housing choices. Similarly, having a place to make friends and connect with tutors from accounts you know are other students will lead to improved wellbeing from larger friend groups and improved academic performance with more people getting tutoring. |

# 3. Feasibility Analysis

## Technical Feasibility

Functionally, our team has a lot of experience being students, as all of us have now attended multiple universities. Additionally, most of us have firsthand experience with the struggle of building a community and finding quality housing when moving to the area from another country. This combined with a broad experience between us of web application building, we expect to be able to develop this relatively small application for the use of Fairfield U students.

* Lots of experience being students
* Experience with the problems face by international students looking for housing and community
* Web application development experience
* Database administration experience
* There are few Fairfield students (potential users) when compared to most web service user count
* Many examples of message board type applications exist

## Economic Feasibility

Developmental costs will be extremely low since all the teams' members aim do well in this course which requires the completion of the application. By being students and developing the application for free, the software will build from will be available for free. With such a small project, the computing can be handled on a home PC, a cost which will be donated by a team member. As the application gains users and credibility, value of potential add space would grow. A second phase of the project would couple hosting ads with paying for the commercial software licensing and commercial server hosting.

* Development occurs for free in class setting
* System components can be brought in for free
* Computing costs are low and able to occur on a home PC
* Ad revenue possible after application becomes established
* Costs would also grow for computing and licensing along with revenue.

## Operational Feasibility

To the benefit of organizational homogeny, the users and developers creating the application are one in the same. We want this application to work well and be accepted by other students because we want to use the information it will contain in our everyday lives. This gives a strong incentive to create an application other students actually want to use. Without other students adding posts and reviews, it will just be an empty shell. Fortunately, other students tend to spend a lot of time looking for housing and community online and are similarly frustrated with current options. Students who have had a bad interaction with a property company are very motived to leave a review and want to post somewhere they know other students will see it. Students looking for community and housing want to know that they are interacting with other real students. If our application provides simple web access to all these things, students are highly likely to use it.

* The organization and the end user have the same needs
* We want the application to succeed because we want to use it
* The students using the app will make or break its usability
* Students have few other trusted options for the same core functionality
* Bad experiences with property companies makes students more likely to want a review platform

# 4.Requirements Definition

## Functional Requirements

### Login

* User visit the Web Application, the default ‘Login Page’ will be prompted
* Email entered by each each user, must be unique and must be Fairfield student email

Eg. [hien.tran@student.fairfield.edu](mailto:hien.tran@student.fairfield.edu)

* Password must be:
* between 8-16 characters long
* standard uppercase characters (A - Z)
* standard lowercase characters (a - z)
* numbers (0 - 9)
* special characters
* Password is not stored as clear text and is stored using latest encryption
* After the user enters the email address, password and clicks ‘Login’ button, email and password will be validated by matching to the database entries.
* If credentials are correct, user will be redirected to ‘Home page’
* If credentials are incorrect, the system will display ‘Invalid email address/password’ in red text

### Forgot Password

* User clicks on ‘Forgot password’, user will be redirected to a ‘Reset Password’ page
* User will be asked to enter his/her email address in the text box
* After the user clicks the 'Send email’ button, the system will match to the database entries
* If email address is correct, a link will be sent to the user’s email address
* If email address is incorrect, the system will display ‘Invalid email address’ in red text
* User opens his/her email address, clicks on the ‘Reset password’ link, user will be redirected to ‘Reset Password’ page
* User will be asked to enter his/her new password manually twice
* A new password must follow the same requirements and must be different from the old password
* If the new password is identical to the old password, the system will display ‘New password cannot be the same as the old password. Please enter a new one’. User will be asked to enter a new password
* If the new password meets all the requirements, the new password will be hashed and saved in the database.
* The system will display ‘Password is reset successfully’
* User will be redirected to the ‘Home Page’

### Create New Account

* User visit the Web Application, the default ‘Login Page’ will be prompted
* User clicks on the ‘Create an account’ button, user will be redirected to ‘Create an Account’ page
* User will be asked to enter his/her student email address in the text box
* The system will validate the email address entered by searching in the database if the account associated with that email address exists or not
* If email address already exists, the system will display ‘Email already exists’, user is asked to re enter his/her valid email address
* If email address is not Fairfield student email address, the system will display ‘Invalid email address’, user is asked to re enter his/her valid email address

* When email address is verified, student is asked to enter his/her password twice
* If the password meets all the requirements, the system will display ‘Account is created successfully’ and account will be sent and saved in the database
* If the password does not meet the requirements, the system will display

           ‘Please enter a valid password that meets the requirements below:

* between 8-16 characters long
* standard uppercase characters (A - Z)
* standard lowercase characters (a - z)
* numbers (0 - 9)
* special characters’
* If the account is created successfully, user will be notified in her email that his/her account is created successfully

### Navigation

* The ‘Home’, ‘Login’ are button that will be clicked
* The 4 sections button includes ‘Housing’. ‘Tutoring’, ‘Market Place’, ‘Social Media’. Users can click on each button and it will redirect to each section page.

       If user clicks on ‘Home’ button, user will be redirected to the ‘Home Page’

* If user clicks on ‘Housing’ button, user will be redirected to the ‘Housing Page’
* If user clicks on ‘Tutoring’ button, user will be redirected to the ‘Tutoring Page’
* If user clicks on ‘Marketplace’ button, user will be redirected to the ‘Marketplace Page’
* If user clicks on ‘Social Media’ button, user will be redirected to the ‘Social Media Page’
* If the user is not logged in then Login will appear in the menu and clicking it will take him/her to the ‘Login Page’
* If the user is logged, Login will be replaced with My Account which will take the user to his/her ‘Profile Page’

### Create post

* User clicks on the section, he/she wants to navigate to, user will be redirected to that section page
* When the user clicks on the ‘Create post’ button, the user will be redirected to the ‘Create a post’ page.
* User will be asked to enter post body in the all the required field
* If user misses any of the required field, when he/she clicks ‘Submit’ button, the system will display ‘Please enter in the required field’ in res text
* Otherwise, all the information entered will be checked and filter out all the inappropriate language.
* After a post is successfully saved into the database, it will be displayed on the ‘Category page’
* User will be notified that his/her post is created successfully and will be redirected to the ‘Category page’ where the post appears

### View/Sort Post

* User clicks on the section that he/she wants to navigate to, user will be redirected to that section page
* On that section page, user clicks on the category that he/she wants to navigate to, user will be redirected to that category page
* All the posts belonged to that category will be displayed in that category page
* Most recent posts will be displayed first, posts are listed in the chronological order
* User clicks on the drop down list and click on the ‘Sort Option’, the system will get posts under selected category by selected sort
* The system will display posts in selected category by sort method
* When user clicks on any posts displayed in the page, user will be redirected to the selected post page
* In the selected post page, all post information will be displayed

### Add Comment

* Under select post page, when user clicks on the ‘leave comment’ button, a comment box will be displayed allowing user to enter his/her comments by text
* After user clicks ‘submit’ button, a comment will be sent and checked and filtered out all inappropriate language
* The comment is saved in the database, the comment will be displayed under the post

## Non-Functional Requirements

### Ease of use

* Once the user logins into the account must be able to perform all the major actions within three clicks.
* The information displayed on the web application should be easily accessible by everyone even if the user is color blind.

### Notifications

* All users get notified prior whenever there is maintenance work going on.
* The user visits the webs application for the first time and creates an account an authentication code is sent to the user's email address within 3 seconds.
* The user gets emails about the new listing on the site and also on the responses from another user this service can be stopped by the user if not required.
* The user gets an update via email that the password has been changed at location and time less than 3 seconds.

### Security

* Password hashing is used to verify the integrity of user passwords whenever a user tries to login into the system.
* At the time of password reset, two-factor authentication is done by sending a code either to the user's email or phone number.
* Users must update their passwords every six months to have more control over their accounts.

### Response Time

* Whenever the user clicks on any component of the webpage or application the reaction time is less than 2 seconds to give more interaction with the user.
* Search action performed by the user will be completed within 3 seconds.
* Whenever the user uses the location services to find the nearby postings, the page refreshes within 3 seconds shows the output.

### Portability

* The system can be easily used on all types of mobile devices and computers irrespective of the screen sizes
* The System should be easily deployable on other technologies like Azure, AWS, and Cisco meta pod.

# 5. Model Driven Architecture

## Model Driven Architecture

Text for model driven architecture

# 6. Functional Model

## Use Case Descriptions

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Name: Create Post | | ID: | Importance Level: High |
| Primary Actor: user | Use Case Type: Detail, Essential | | |
| Stakeholders and Interests:  User – wants to make a post | | | |
| Brief Description: To add content to the site, Users make Posts. They navigate to the Section that they want their post to appear in and are prompted to enter specific and general information depending on the Section they are posting in. | | | |
| Trigger:  Type: External - User initiated | | | |
| Relationships:  Association: Student  Include:  Extend: Review a Landlord  Generalization: Manage listings | | | |
| Normal Flow of Events:   1. An authenticated User navigates to the posting section they would like their post to appear in 2. The User clicks the “Add a Post” Button at the top of the page 3. The User is taken to a new page where they must enter a Title and their desired text in a large box for the body of their post If the User is in the Housing section   The “Review a Landlord” use case is executed  If the User is in the Tutoring Section  The S-2: Tutoring post sub flow is performed  If the User is in the Social section  The S-3: Social Post sub flow is performed If the User is in the Marketplace Section  The S-4: Marketplace Post sub flow is performed   1. The user is given an “Add Photos button for their post, the first of which will be the post Thumbnail If no photo is uploaded a default thumbnail will be given basted on Category 2. System places a preview of the uploaded photos on the in-work post 3. The User Clicks the “Submit post button” 4. The system validates that all required fields are populated If required fields are not populated the user is left on the page and the missing fields highlighted 5. The System Creates a post under the specified Section and Category 6. The User is given a notification that their post was successfully created 7. The user is directed to the Category Page of the Section their post appears in | | | |
| SubFlows:   1. Housing Post  1. A dropdown is given for the User to select whether they have an open room or are looking for one 2. System adds additional required fields to be filled in (See Appendix: Housing Post) 2. Tutoring Post:   1. A dropdown menu is given for the User to select whether they are a Tutor or Tutoree  2. The system adds additional required fields to be filled in (see Appendix: Tutoring post requirements)  3.   1. Social Post:   1. A dropdown is given for the User to select the category of the post e.g., Outdoors, Video Games, Hobbies, Social Event, Concerts/Live Music. (See Appendix: Social Post requirements)  2. The system adds additional required fields to be filled in e.g., location, date and time, reoccurring? (See Appendix: Social post requirements)   1. Marketplace Post:   1. A dropdown menu is given for the User to select whether they are Selling or Wanting to Buy  2. The system adds additional required fields to be filled in e.g., price, location, (See Appendix: Market Place Post requirements)  Cancel Post Creation  1.The user may click a “Discard Post” button on the in work post at any time to cancel the post creation  2. The user is returned to the Section Page they started on. | | | |
| Alternate/Exceptional Flows:  Cancel Post Creation  1.The user may click a “Discard Post” button on the in work post at any time to cancel the post creation  2. The user is returned to the Section Page they started on. | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Name: View and Sort Posts | | ID: | Importance Level: High |
| Primary Actor: (trigger of the use case)  User | Use Case Type: (overview or detail and essential or real) Detail, Essential | | |
| Stakeholders and Interests:  (All actors that have interest in use cases. Must include primary actor)  Student - wants to sort rental listing by date, price, distant | | | |
| Brief Description: This use case describes how the listing is sorted by date, price and distant | | | |
| Trigger:  Type: (external or temporal) External | | | |
| Relationships:  Association: Student  Include:  Extend: Forgot Password  Generalization: Manage listings | | | |
| Normal Flow of Events:   1. An authenticated User logs in and is on the Home Page 2. The User clicks one of four buttons labeled with each Section of the application 3. The System takes the User to a page with buttons for each of the Categories under that Section 4. The user clicks the button corresponding with the Category of posts they would like to view 5. The System takes the User to a page listing the posts within that category by recently posted 6. The user is given a dropdown menu to alter the sorting of the posts if desired  If the User selects the dropdown menu   The S-1: Sort Posta sub flow is performed   1. The User clicks on the post that they want to view 2. The System takes the User to the post they selected | | | |
| SubFlows:  S-1 Sort Posts:   1. If the user is in the Housing Section, the dropdown is populated with the Sortable fields from Appendix: Housing Posts If the User is in the Tutoring Section, the dropdown is populated with the Sortable fields from Appendix: Tutoring Posts If the User is in the Social Section, the dropdown is populated with the Sortable fields from Appendix: Social Posts If the User is in the Marketplace Section, the dropdown is populated with the Sortable fields from Appendix: Marketplace Posts 2. The system adds standard sorting options to the dropdown, such as “Alphabetical” 3. The User selects the sorting option they desire from the dropdown 4. The System sorts the posts displayed from the Category by the selected sort parameter | | | |
| Alternate/Exceptional Flows: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Name: Manage a Post | | ID: | Importance Level: High |
| Primary Actor: User | Use Case Type: Detail, Essential | | |
| Stakeholders and Interests:  User – wants to edit or delete a post they made | | | |
| Brief Description: | | | |
| Trigger:  Type: External | | | |
| Relationships:  Association: My Posts  Include:  Extend:  Generalization: Manage Post | | | |
| Normal Flow of Events:   1. A User logs into the system and wants to edit or delete on of their posts 2. User navigates to the desired post Execute Use Case: View and Sort Posts or Use Case: View My Posts 3. If the User clicks the “Edit” Button S-1: Edit Post is performed  If the user clicks the Delete button S-2: Delete Post is Performed 4. User is returned the category list their post is/was in | | | |
| SubFlows:  S-1: Edit Post:   1. The System takes the User to a page configured the same as the “Create Post” page, with all the information from the post in the fields 2. The User makes changes to any of the information except the post “Title” 3. The User clicks the “Submit Changes” button at the bottom of the page when done making changes 4. The System Validates that all required fields are still populated If required fields are left blank, the User is kept on the page and the missing fields highlighted 5. The System updates the post 6. The user receives notification that the edit was successful   S-2: Delete Post:   1. The user is given a popup to confirm that they wish to delete the post 2. The user selects clicks a button to confirm 3. The System Deletes the post 4. The User is given notification that the post was deleted | | | |
| Alternate/Exceptional Flows: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Name: Review a Landlord | | ID: | Importance Level: High |
| Primary Actor: User | Use Case Type: Detail, Essential | | |
| Stakeholders and Interests:  User – Creates landlord reviews | | | |
| Brief Description: User wants to leave a review of a landlord | | | |
| Trigger:  Type: External | | | |
| Relationships:  Association: Existing User  Include:  Extend:  Generalization: Post | | | |
| Normal Flow of Events:  1. A User logs in and wants to leave a review of a landlord  2. User navigates to the Landlords category  See Use Case: View and Sort Posts  3. User Selects a Landlord they would like to leave a review for  If desired Landlord does not exist S-1: Request New Landlord sub flow is followed  4. User is taken to the page listing reviews for the selected Landlord  5. User clicks “Leave a Review” Button at the top of the page  6. User is taken to a page where they are given fields to add the title, post body, 1-5 star rating, photos and other required fields (See Appendix: Landlord Review Posts)  7. User clicks the “Submit Review” button  8. System validates that all required fields are filled in  If not, the user remains on the page and the missing fields are highlighted  9. System creates a new review under the appropriate Landlord  10. User is given a notification that their review has been posted  11. User is returned to the listing of reviews where their post was created. | | | |
| SubFlows:   1. Request New Landlord 1.User clicks “Add a Property” Next to the “Leave a Review Button,  2. User is taken to a page where information about the Landlord can be Entered e.g., Name, type, company logo (See Appendix: Landlords) 3. User clicks “Submit Request” button 4.System validates that all required fields are populated  If not, the user remains on the page and the missing fields are highlighted 5.The System adds the request to the landlord request queue 6. The User is notified that their request was submitted and is returned to the list of Landlords | | | |
| Alternate/Exceptional Flows: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Name: Create User Account | | ID: | Importance Level: High |
| Primary Actor: User | Use Case Type: | | |
| Stakeholders and Interests:  User – Allowed to create a new account | | | |
| Brief Description: | | | |
| Trigger:  Type: User visits application and initiates user account creation | | | |
| Relationships:  Association: New User, Existing User  Include:  Extend: Create User Account  Generalization: Create User | | | |
| Normal Flow of Events:   1. A New User visits the public facing homepage for the application 2. The user selects the “Create New Account” button 3. The user is taken to a page with fields to enter their User Account Information, including student email where information is saved temporarily to a New User 4. The system validates the information provided for User Account Information - Use case Validate New User occurs  - If a user enters invalid information, the S-1 subflow is followed 5. The information given for Account information is saved by the system, creating an Existing User account 6. The user is automatically logged in and returned to the home page 7. The use case ends. | | | |
| SubFlows:  S-1: User Enter Invalid Information  If during Create Account, the system determines that the User entered invalid User  Account information, the following occurs:  1. The system describes which entered data was invalid and presents the User with suggestions  for entering valid data.  2. The system prompts the User to re-enter the invalid information.  3. The User re-enters the information and the system re-validates it.  4. If valid information is entered, the User Account Information is stored.  5. If invalid information is entered, the Entered Information is Invalid alternative flow is  executed again. This continues until the User enters valid information or chooses Cancel  (see the User Cancels Account Management Request alternative flow). | | | |
| Alternate/Exceptional Flows:  User Cancel Request  At any time, the User may choose to cancel the account creation. At which point,  the processing is discontinued, the user account remains unchanged, and the user is  notified that the account management request has been cancelled. | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Name: Log Into Application | | ID: | Importance Level:High |
| Primary Actor: User | Use Case Type: Detail, Essential | | |
| Stakeholders and Interests:  User – Wants to access application | | | |
| Brief Description: A user visits the application homepage and logs in | | | |
| Trigger:  Type: (external or temporal) | | | |
| Relationships:  Association: Existing User  Include:  Extend: Reset Password  Generalization: User Authentication | | | |
| Normal Flow of Events:   1. User navigates to public facing Hope Page 2. User click “Login” Button 3. User is taken to a page to enter their student email and password 4. User clicks “Submit” 5. System validates that username and password match If not, user is kept on page, given a notification their attempt was unsuccessful, and provided a link to reset their password (See Use Case: Reset Password) 6. User is directed to internal homepage | | | |
| SubFlows:   1. (Events that can take place during the normal flow of events) | | | |
| Alternate/Exceptional Flows:  Cancel Login:   1. User clicks “Cancel” button while on logon page 2. User is redirected to public home page 3. Use case terminates | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Name: Request Direct Message Conversation | | ID: | Importance Level: High |
| Primary Actor: User | Use Case Type: Detail, Required | | |
| Stakeholders and Interests:  User- allowed to send a request to the other users on the site. | | | |
| Brief Description: A User requests a direct message textual conversation with another user that has already made a post. | | | |
| Trigger:  Type: User Sends a request to another user | | | |
| Relationships:  Association: Existing User  Include:  Extend: Use Case: Direct Message Conversation  Generalization: Message | | | |
| Normal Flow of Events:   1. User logs in and navigates to a post left by another user  See Use Case: View and Sort Posts 2. User clicks “DM user button” 3. User is prompted to confirm that they want to request a direct message conversation with the other user 4. User clicks the confirmation button If user clicks “Cancel” button they are returned to the post they were viewing 5. System sends request to Message box of User2 6. User2 accepts Conversation request by clicking “Accept” Button in message If User2 clicks “Decline” Alternate flow: Deney Request is performed 7. A direct message chain is created in User’s Message Box 8. Users can now converse Execute use Case: Direct Message Conversation | | | |
| SubFlows: | | | |
| Alternate/Exceptional Flows:  Deney request:   1. System sends User a notification that their request weas denied 2. Parent flow ends | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Name: Leaving a Comment | | ID: | Importance Level: High |
| Primary Actor: User | Use Case Type: Update, Functional | | |
| Stakeholders and Interests:  -User- Wants to leave feedback on a post | | | |
| Brief Description: User leaves a comment on a post or review. | | | |
| Trigger:  Type: User adds a comment to a post | | | |
| Relationships:  Association: Existing User, Post  Include:  Extend:  Generalization: Comment | | | |
| Normal Flow of Events:   1. A User logs in and navigates to the post or review they would like to leave a comment on See Use Case: View and Sort Posts 2. User scrolls down past the previous left comments, if there are any, and clicks the leave a comment button. 3. User is presented with a box where they enter their comment 4. User clicks “Submit” button  If user clicks “Cancel Button” Alternate Flow Cancel Comment is followed 5. User is returned to the post to view their new comment | | | |
| SubFlows: | | | |
| Alternate/Exceptional Flows:  Cancel Comment:   1. If text has already been entered, User is given a popup asking them to confirm they want to cancel comment  If they do not confirm, they are able to continue with their comment 2. The System clears the text entered 3. The user no longer sees the comment text box or its submit/cancel buttons on the post | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Name: Reset Password | | ID: | Importance Level:High |
| Primary Actor: User | Use Case Type: Detail, Essential | | |
| Stakeholders and Interests:  User – Wants to reset their password | | | |
| Brief Description: A user logs in and views, responds to, or deletes a message | | | |
| Trigger:  Type: external | | | |
| Relationships:  Association: Existing User  Include:  Extend:  Generalization: User Authentication, Update Account Information | | | |
| Normal Flow of Events:   1. User navigates to public facing Hope Page 2. User click “Login” Button 3. User is taken to a page to enter their student email and password 4. User clicks “Reset Password” button below the “Submit” and “Cancel” buttons 5. User is taken to a new page with a field to enter their student email address 6. User clicks “Send reset email” button 7. System validates that email belongs to an Existing User If not, user remains on page and is given notification of the mismatch 8. System sends reset link to student email address 9. User clicks on the link and is taken to a page to enter their new password twice 10. User clicks “Set Password” button 11. System confirms that both fields match If not, user is kept on the page with a notification of the mismatch 12. User is given notification that the update was successful and taken to their homepage | | | |
| SubFlows: | | | |
| Alternate/Exceptional Flows:  Cancel Login:   1. User clicks “Cancel” button while on submit email page 2. User is redirected to public login page 3. Use case terminates | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Name: Map view | | ID: | Importance Level: |
| Primary Actor: User | Use Case Type: | | |
| Stakeholders and Interests:  -User- can view the posts that are nearby | | | |
| Brief Description:  Allowing User to look up for the posts around him. This is done by using location service. | | | |
| Trigger:  Type: User clicks on the nearby tab | | | |
| Relationships:  Association: Existing User  Include: User account login  Extend:  Generalization: | | | |
| Normal Flow of Events:   1. User clicks on the nearby tab which is present on the right side of search section. 2. User will be asked to enable location to use this feature. 3. If the User enables the location service   The S-1: Map window opens up   1. If the User disable the location service   The S-2: Error message pops up | | | |
| SubFlows:   1. Map shows the result   User must type in the location that he is in or interested  User must provide the radius with in the location he is looking for.  System generates a search ID  System shows the post based on location services  S-2: Error message pops up  System generates a declined location request ID  User will be asked to enable location services to search the posts nearby | | | |
| Alternate/Exceptional Flows: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Name: View inbox | | ID: | Importance Level: |
| Primary Actor: User | Use Case Type: | | |
| Stakeholders and Interests:  -User- can access all notifications | | | |
| Brief Description:  Helps User to check for the notifications from post, ratings, reviews related to him at one point. | | | |
| Trigger:  Type: User clicks on the notification tab | | | |
| Relationships:  Association: Existing User  Include: User account login  Extend:  Generalization: | | | |
| Normal Flow of Events:  1. Notifications get piled up and the number of unopened notifications will be shown  2. User clicks on the notification tab on the navbar section  3. Opens a window which show the response for message request  4. Notifications are sorted by using time and date they have been received.  5. The System stores the entire data in the database so that the user can check for the notifications received a long time ago. | | | |
| SubFlows: | | | |
| Alternate/Exceptional Flows: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Name: Report a post | | ID: | Importance Level: |
| Primary Actor: User | Use Case Type: | | |
| Stakeholders and Interests:  -User- can report a post | | | |
| Brief Description:  Allows User to report a post that he found it was spreading False Information, Violence  and Spam. This helps the admin to discard the post | | | |
| Trigger:  Type: User Clicks on report icon available on the post | | | |
| Relationships:  Association: Existing User  Include: User account login   |  |  |  |  | | --- | --- | --- | --- | | Use Case Name: Adding user details in profile | | ID: | Importance Level: | | Primary Actor: User | Use Case Type: | | | | Stakeholders and Interests:  -User- adds on more details in profile | | | | | Brief Description:  User can add details like personal interests, hobbies, profile picture, contact number and profile picture. All these field are optional and this helps for other users to know about the person. | | | | | Trigger:  Type: User Clicks on edit profile. | | | | | Relationships:  Association: Existing User  Include: User account login  Extend:  Generalization: Verification | | | | | Normal Flow of Events:  1. User clicks on the profile, finds his data that entered at the time of account creation.  2. Then clicks on the edit section in profile  3. Then a window pops up which contains field like hobbies, personal interests, add address and  add contact number.  4. User can upload a picture in profile section.  5. User can enter the details in the above field in which interested.  6. User clicks on update button.  7. A message shows up stating that details are updated. | | | | | SubFlows: | | | | | Alternate/Exceptional Flows: | | | |   Extend:  Generalization: Report | | | |
| Normal Flow of Events:  1. User checks on the new posts available on the site  2. User finds the post is not related to housing or rental or group study events  3. User clicks on the report section available on the post  S-1: Report post is performed  4. User selects the close arrow  5. User gets back to view post session | | | |
| SubFlows:  S-1: Report Post:  1. User gets a new window opened up to select the problem he is facing  2. User selects any one of the option available  3. User then clicks on the submit button  4. A message pops stating that the post was reported and waiting for the review from the  admin. | | | |
| Alternate/Exceptional Flows: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case Name: Validating a New User | | ID: | Importance Level: High |
| Primary Actor: User | Use Case Type: | | |
| Stakeholders and Interests:  -User- account gets verified | | | |
| Brief Description:  User account get verified by the system at the time of account creation by sending a  verification code to the user email provided at the time of creating account. | | | |
| Trigger:  Type: User Clicks on create account | | | |
| Relationships:  Association: New User  Include:  Extend:  Generalization: Verification | | | |
| Normal Flow of Events:  1. User enters his information and also his student email at the time of account creation  2. System checks if the email id entered by the user matches with the pattern  provided(@student.fairfield.edu)  3. User clicks on create account  4. A new window pops up showing enter the code sent to your email. This code is valid for 10  minutes  5. If User enters correct code  S-1 sub flow is followed  If User enters wrong code  S-2 sub flow is followed | | | |
| SubFlows:      1.A message like account created successfully is displayed.  2.An email is also sent to the user that his account is created successfully  S-2:  1. An error pops up stating that entered code is wrong  2. User can enter the code again or click on send a new code  3. A new authentication code is sent to the user. | | | |
| Alternate/Exceptional Flows: | | | |

# 7. Structural Model

## CRC Cards

### Student

Front:

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name: Student | ID: 0 | | Type: Concrete, domain |
| Description: a Fairfield University Student | | | Associated Use Cases: 1 (user) |
| Responsibilities | | Collaborators | |

Back:

|  |
| --- |
| Attributes:  Student ID  User ID  Name  Gender  Email  Phone |
| Relationships:  Generalization (a-kind-of): Person  Aggregation (has-parts):  Other Associations: |

### User

Front:

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name: User | ID: 1 | | Type: Concrete, Domain |
| Description: Student who creates a post | | | Associated Use Cases: Post (2), |
| Responsibilities | | Collaborators | |

Back:

|  |
| --- |
| Attributes:  User ID  Student ID  User name  Passwords  Email |
| Relationships:  Generalization (a-kind-of): Person  Aggregation (has-parts):  Other Associations: |

### Post

Front:

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name: Post | ID: 2 | | Type: Abstract, Domain |
| Description: Post that is created by User | | | Associated Use Cases: 1 (User), 3 (thumbnails), 5 (section), 6 (category) 4 (notification) |
| Responsibilities | | Collaborators | |

Back:

|  |
| --- |
| Attributes:    Post ID  User ID  Category ID  Title  Description  Date |
| Relationships:  Generalization (a-kind-of): Post  Aggregation (has-parts):  Other Associations: User, section |

### Thumbnail

Front:

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name: Thumbnail | ID: 3 | | Type: Abstract, Domain |
| Description: Picture that User uploads | | | Associated Use Cases: 2 (post) |
| Responsibilities | | Collaborators | |

Back:

|  |
| --- |
| Attributes:  Thumbnail ID  Post ID  Name  Size |
| Relationships:  Generalization (a-kind-of): Post  Aggregation (has-parts):  Other Associations: |

### Notification

Front:

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name: Notification | ID: 4 | | Type: Abstract, Domain |
| Description: Notification that User receives when post is created successfully | | | Associated Use Cases: 1 (User), 2 (post), 8 (review), 10 (landlord request) |
| Responsibilities | | Collaborators | |

Back:

|  |
| --- |
| Attributes:  Notification ID  Post ID  Review ID  Request ID  User ID  Message |
| Relationships:  Generalization (a-kind-of): Post  Aggregation (has-parts):  Other Associations: |

### Section

Front:

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name: Section | ID: 5 | | Type: |
| Description: Section  eg Housing, Tutoring, Marketplace, Social | | | Associated Use Cases: 6 (category) |
| Responsibilities | | Collaborators | |

Back:

|  |
| --- |
| Attributes:  Section ID  Name  Description |
| Relationships:  Generalization (a-kind-of): Post  Aggregation (has-parts):  Other Associations: |

### Category

Front:

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name: Category | ID: 6 | | Type: |
| Description: Category in each Sections | | | Associated Use Cases:  5 (section), 2 (post) |
| Responsibilities | | Collaborators | |

Back:

|  |
| --- |
| Attributes:  Category ID  Section ID  Name |
| Relationships:  Generalization (a-kind-of):  Aggregation (has-parts):  Other Associations: |

### Landlord

Front:

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name: Landlord | ID: 7 | | Type: Concrete, Domain |
| Description: Landlord that is reviewed by User | | | Associated Use Cases:  8 (review), 10 (request) |
| Responsibilities | | Collaborators | |

Back:

|  |
| --- |
| Attributes:  Landlord ID  Name  Type  Phone  Email  Address  Rating |
| Relationships:  Generalization (a-kind-of):  Aggregation (has-parts):  Other Associations: |

### Review

Front:

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name: Review | ID: 8 | | Type: Abstract, Domain |
| Description: Review on Landlord left by Users | | | Associated Use Cases: 1 (User), 7 (landlord) |
| Responsibilities | | Collaborators | |

Back:

|  |
| --- |
| Attributes:  Review ID  Landlord ID  User ID  Title  Content  Rating |
| Relationships:  Generalization (a-kind-of):  Aggregation (has-parts):  Other Associations: |

### Landlord Request

Front:

|  |  |  |  |
| --- | --- | --- | --- |
| Class Name: Landlord request | ID: 10 | | Type: Abstract, Domain |
| Description: new landlord is requested by User | | | Associated Use Cases: 1 (User), 7 (landlord) |
| Responsibilities | | Collaborators | |

Back:

|  |
| --- |
| Attributes:  Request ID  User ID  Name of Landlord  Type |
| Relationships:  Generalization (a-kind-of):  Aggregation (has-parts):  Other Associations: |

## Class Diagrams

Create User Account

Diagram

Description automatically generated

SIGNIN

Diagram

Description automatically generated

Edit/Delete Account

Diagram

Description automatically generated

Create Admin Account

Diagram

Description automatically generated

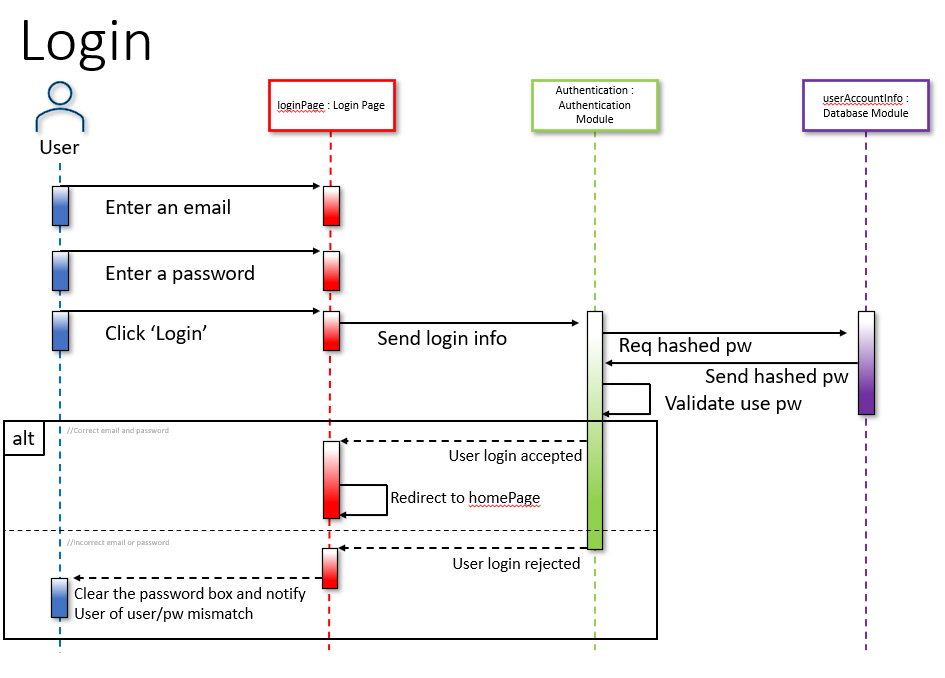
Review Landlord

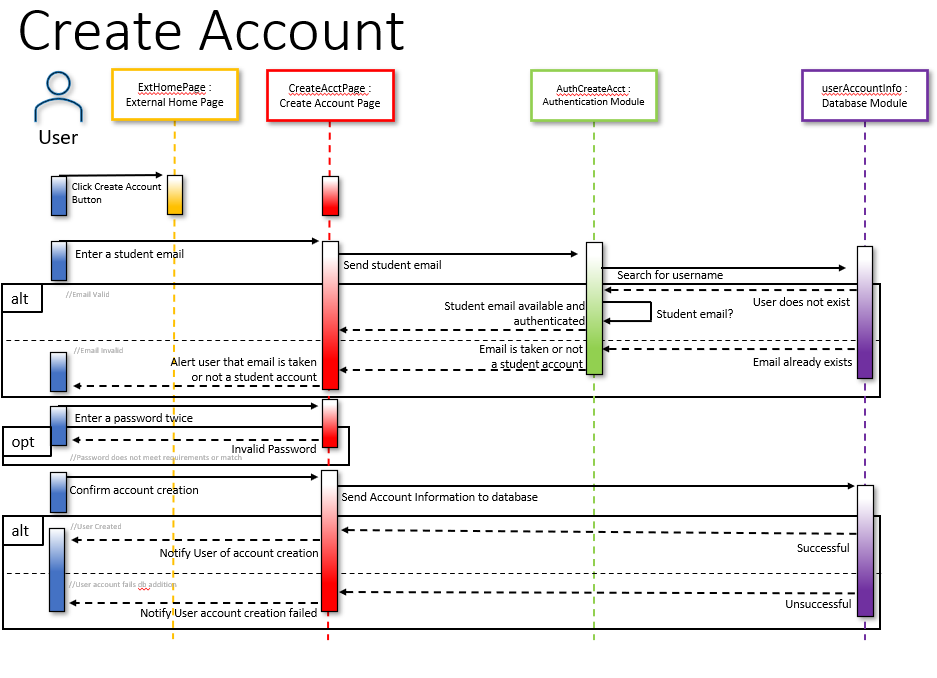
Diagram

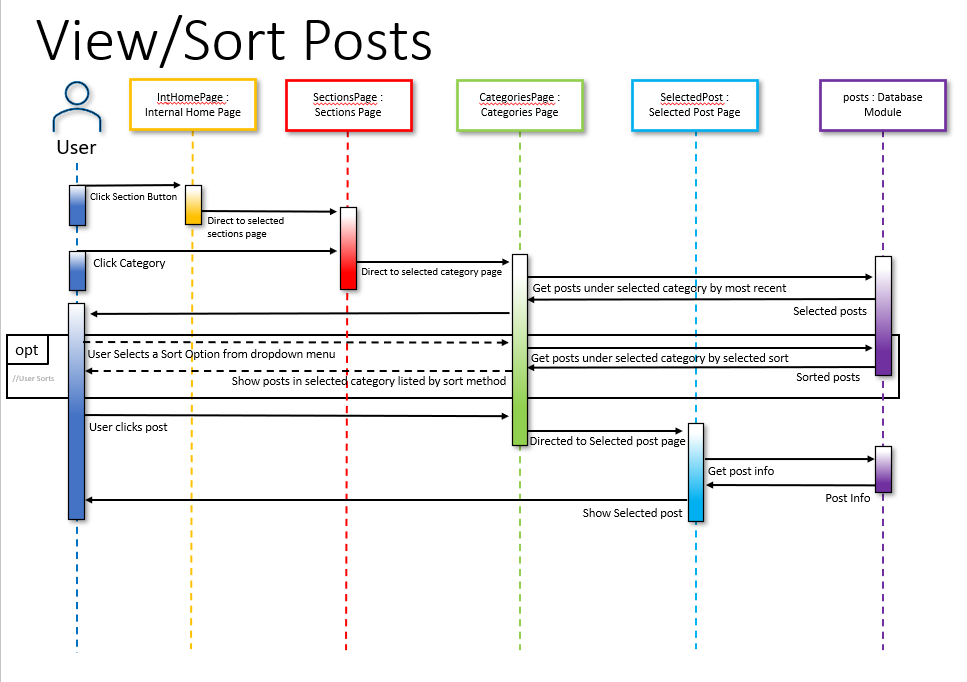
Description automatically generated

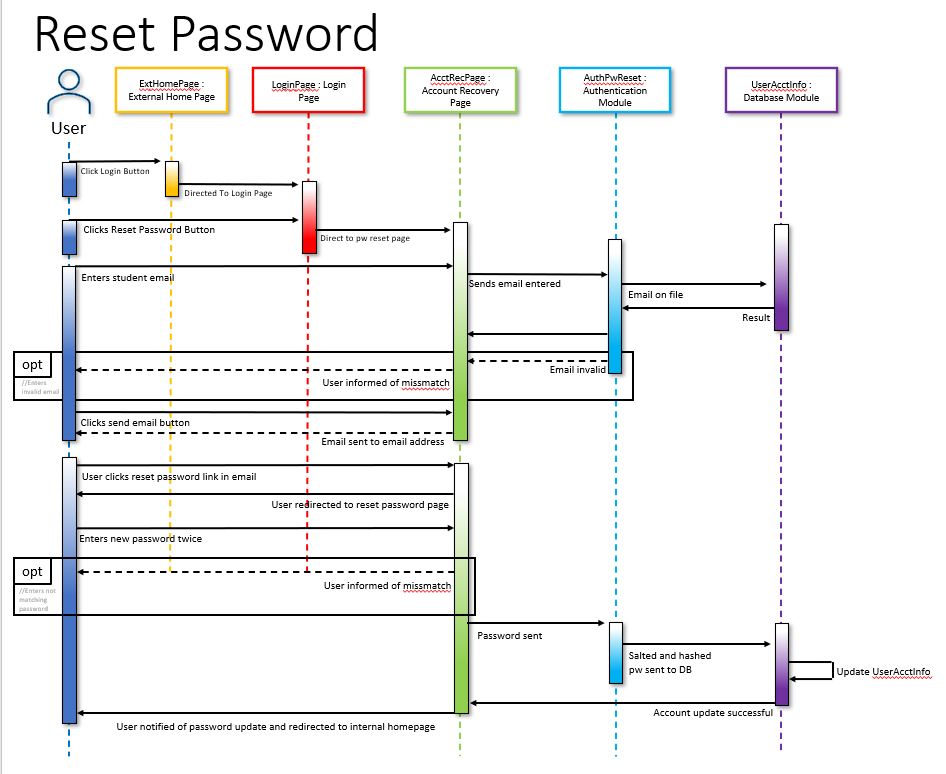
# 8. Behavioral Model

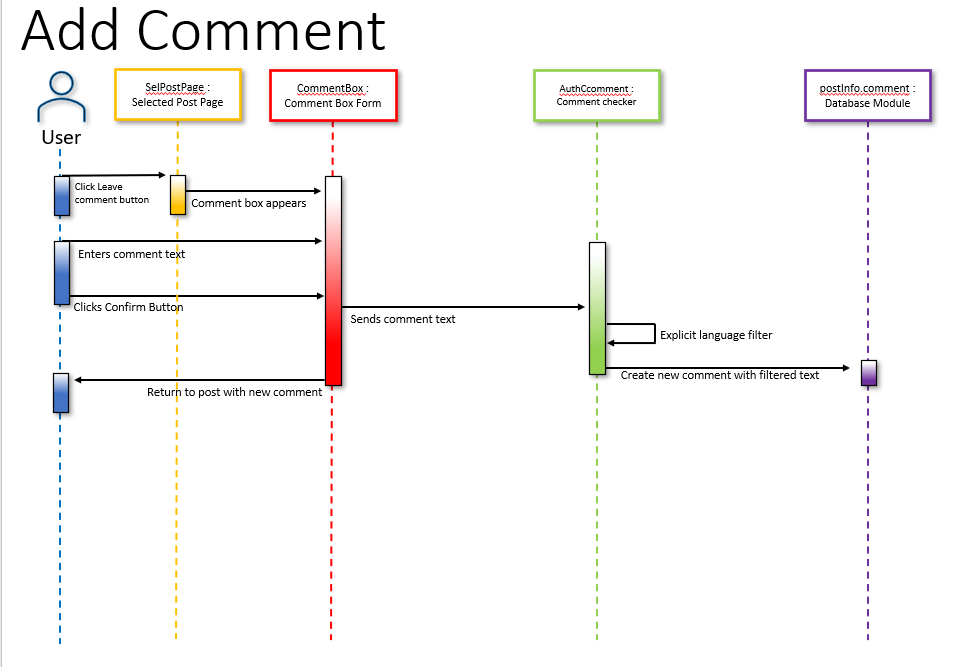
## Sequence Diagrams

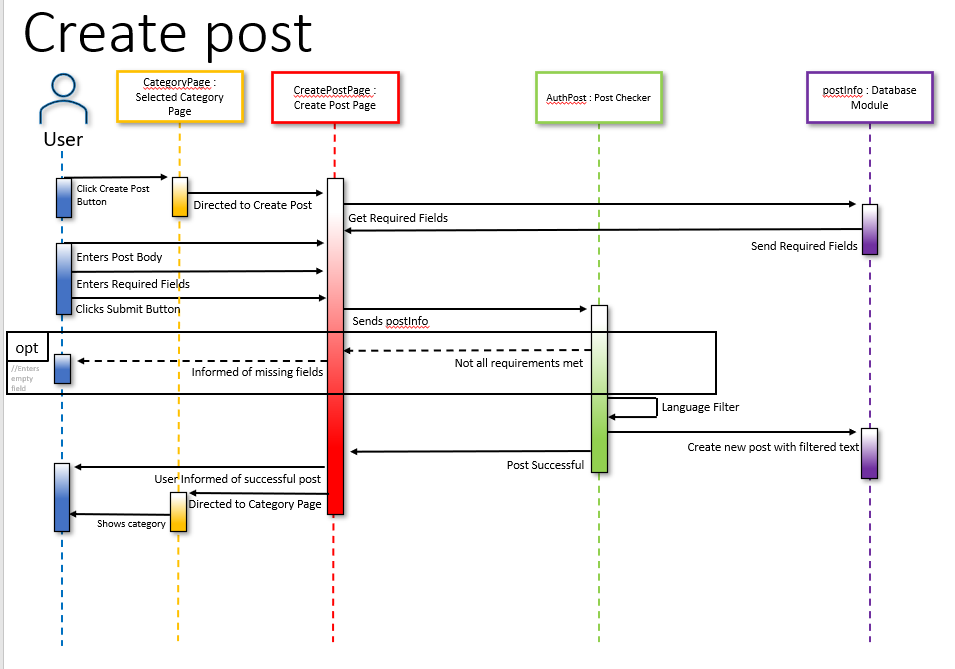


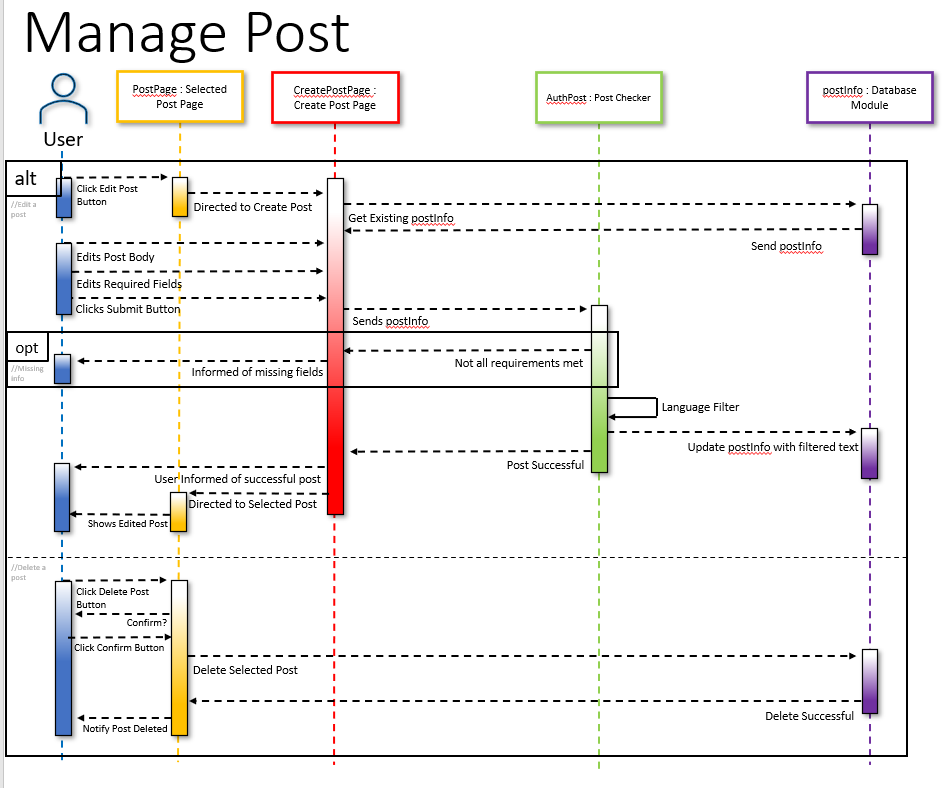


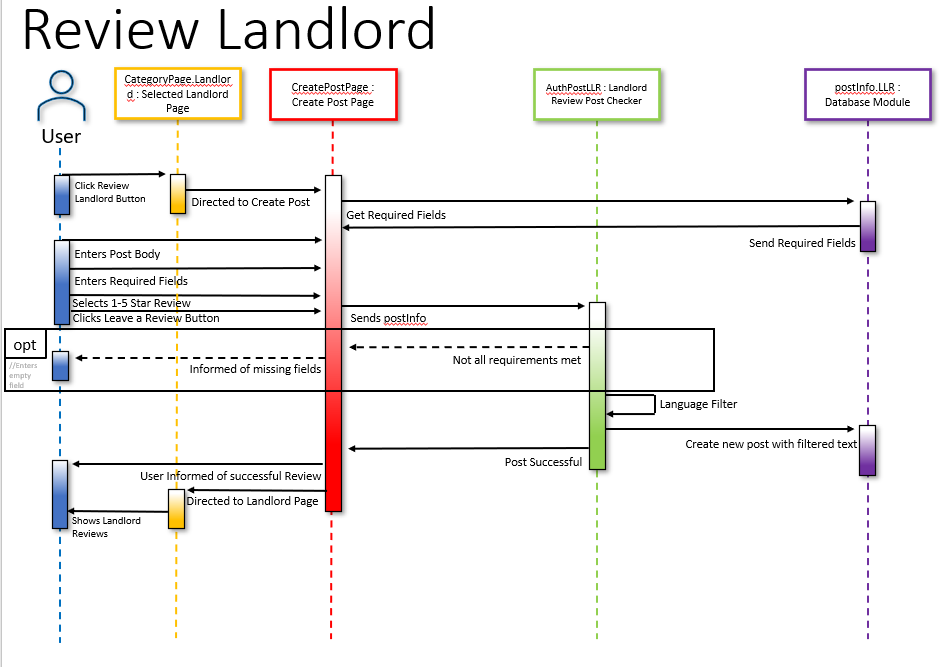


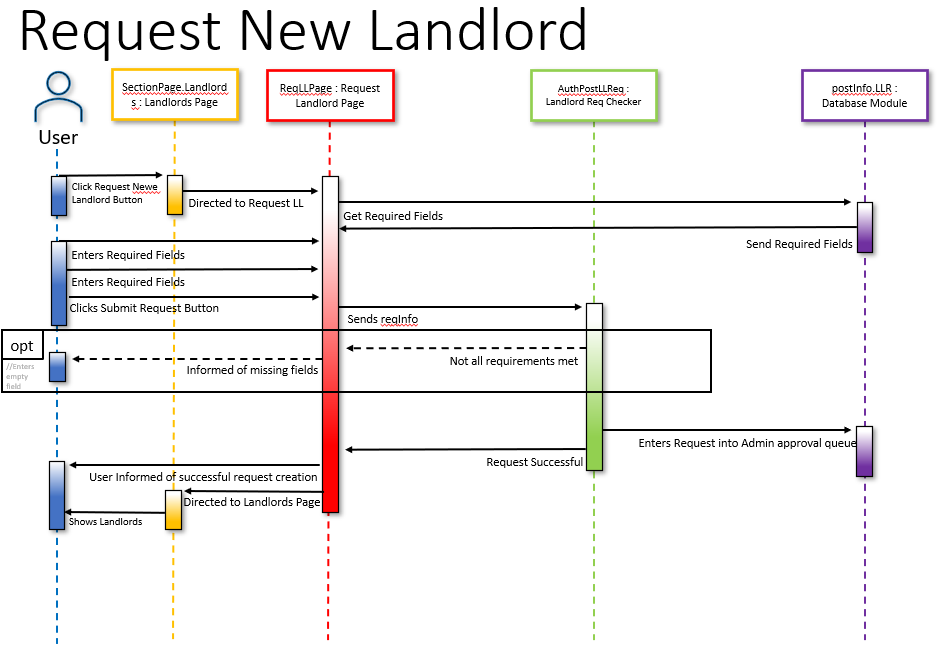


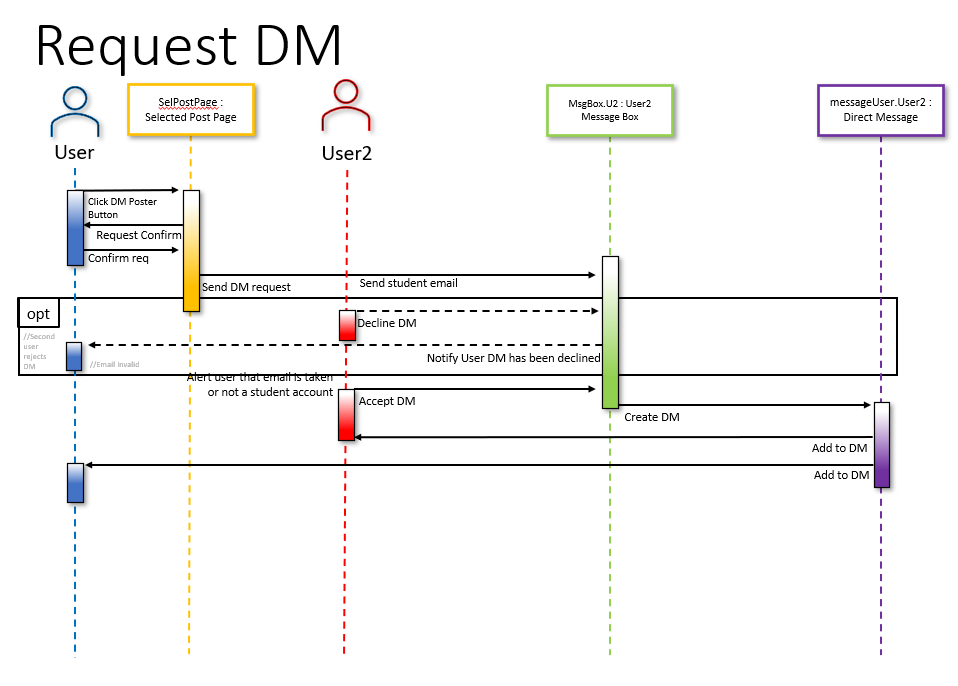












## State Diagrams

### Create User Account:

Diagram

Description automatically generated

### User Sign In :

Diagram

Description automatically generated

### Edit or Delete:

Diagram

Description automatically generated

### Review Land Lord:

Diagram

Description automatically generated

### Create Admin Account:

Diagram

Description automatically generated

# 9. Prototype

## Prototype Screenshots

### Public Home Page

Graphical user interface, website

Description automatically generated

### Sign In Page

A picture containing text, tree, outdoor, grass

Description automatically generated

### Sign In (Failed)

A picture containing text, tree, outdoor, grass

Description automatically generated

Create Account A picture containing text, tree, grass, outdoor

Description automatically generated

### Successful Account Creation

A picture containing text, tree, outdoor, grass

Description automatically generated

### Account Verified

A picture containing text, tree, outdoor, grass

Description automatically generated

### Create Account Errors

A picture containing text, tree, outdoor, grass

Description automatically generated

### Forgot Password

A picture containing text, tree, outdoor, grass

Description automatically generated

### Set New Password

A picture containing text, tree, outdoor, grass

Description automatically generated

### Confirm New Password

A picture containing text, tree, outdoor, grass

Description automatically generated

### Internal Home Page

Graphical user interface, website

Description automatically generated

### Admin Home Page

A picture containing text, sign, screenshot

Description automatically generated

### Community Page

A group of people on a beach

Description automatically generated with medium confidence

### Tutoring Page

A picture containing text, person

Description automatically generated

### Marketplace Page

A picture containing text, person

Description automatically generated

### Housing Page

A picture containing text, outdoor, sign

Description automatically generated

### Landlords Page

Graphical user interface, text, application

Description automatically generated

### Sorting Landlords

Graphical user interface, text, application

Description automatically generated

### Housing Reviews

Table

Description automatically generated

### Sorting Housing Reviews

Table

Description automatically generated