**System Requirements Specification**

Piece of Eden Rentals

4/26/2016

Customer: John Winder

Team 3:

Rachel Cohen

Stephen Masterson

Eric Forte

Matthew Walker

Nicholas Keckeisen

Table of Contents

1. Introduction

1.1 Purpose of This Document

1.2 Purpose of the Product

1.3 Product Scope

2. Functional Requirements

2.1 Use Case 1

2.2 Use Case 2

2.3 Use Case 3

2.4 Use Case 4

2.5 Use Case 5

2.6 Use Case 6

2.7 Use Case 7

2.8 Use Case 8

2.9 Use Case 9

3. Use Case Tests

3.1 Use Case 1 Test - Searching the Database

3.2 Use Case 2 Test - Viewing Detailed Music Meta-data

3.3 Use Case 3 Test - Favorite An Item

3.4 Use Case 4 Test - Rating An Item

3.5 Use Case 5 Test - Reviewing An Item

3.6 Use Case 6 Test - Registering to the Database

3.7 Use Case 7 Test - Logging in to the Database

3.8 Use Case 8 Test - Viewing User Profile

3.9 Use Case 9 Test – Register User Account

3.10 User Case 10 Test - Calendar

4. Non-Functional Requirements

5. User Interface

6. Deliverables

7. Open Issues

8. Appendix A – Agreement Between Customer and Contractor

9. Appendix B – Team Review Sign-off

10. Appendix C – Document Contributions

**1. Introduction**

Welcome to the Piece of Eden Rentals System Requirements Specifications. Piece of Eden Rentals is a beachfront property rental web application, which allows users to put their beachfront properties up for rent, choose properties to rent, and rate other properties.

* 1. **Purpose of This Document**

This document describes how to use the Piece of Eden Rentals web application, as well as the functions and features of the product. The audience for this document is the development team (Team 3) and the customer (John Winder).

* 1. **Purpose of the Product**

Piece of Eden was designed to provide a service to rent and list beachfront properties all across the world. Users of the product will be able to create a user profile, list a property, search for a 3 property, and rent a property for a period of time.

* 1. **Product Scope**

The Piece of Eden web application consists of multiple use cases including but not limited to: registering a property, clicking on a property pin, rating a property, logging in to a user account, renting a property, selecting a user avatar, viewing a user page, visit a webpage, and create a user. Please refer to the rest of the document for further understanding of use cases.

Figure 1. Use Case Diagram

User Database

Register a Property

User Database

Rate a Property

Delete Properties

Click on a Property Pin

**User**

User Database

Select a User Avatar

Approve Properties

Log in to User Account

View User Page

Rent Property

Create a User

Rental Calendar

Visit Homepage

**Admin**

**2. Functional Requirements**

**2.1 Use Case 1**

|  |  |  |
| --- | --- | --- |
| **Number** | 1 | |
| **Name** | Register Property | |
| **Summary** | Register a property for rent. Can include name, location, description and pictures | |
| **Priority** | 5 | |
| **Preconditions** | User has an account | |
| **Postconditions** | The property will be viewable and rentable by other users | |
| **Primary Actor** | Property Owner | |
| **Secondary Actors** | Google API, Property Database | |
| **Trigger** | Selecting the “Register property” button | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User selects the “Register property” button |
|  | 2 | Using Google Maps API the user can select location |
|  | 3  4  5 | User is prompted to enter a Name, Description and Pictures of the property  User selects the “Register” button  The property is added to the database and should be viewable to other users |
| **Extensions** | **Step** | **Branching Action** |
|  | 2a  2b  4a  4b  3a  3b | User Select “Cancel” button  User is returned to the home screen  User Select “Cancel” button  User is returned to the home screen  User does not upload pictures  Continues normally with no pictures for the property |
| **Open Issues** |  | |

**2.2 Use Case 2**

|  |  |  |
| --- | --- | --- |
| **Number** | 2 | |
| Name | Click on property pin | |
| **Summary** | User clicks on the property’s google map pin | |
| **Priority** | 5 | |
| **Preconditions** | A property is registered and displayed on the map | |
| **Postconditions** | Window displays (image of house, house style (modern, cottage, etc), availability windows,price, star rating for property | |
| **Primary Actor** | User | |
| **Secondary Actors** | Property Database | |
| **Trigger** | Clicking on property pin | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User clicks on property pin |
|  | 2 | Window opens and displays the property information from the database |
|  | 3 | User can choose to rent property, or close window |
| **Extensions** | **Step** | **Branching Action** |
|  | 3a  3b | If the user chooses to rent the property, will be taken to a renting property window  If the user chooses to close the window, they will be returned to the map |
| **Open Issues** |  | |

**2.3 Use Case 3**

|  |  |  |
| --- | --- | --- |
| **Number** | 3 | |
| Name | Rate property | |
| **Summary** | Rate a property. The property’s rating will move up or down depending on the rating | |
| **Priority** | 5 | |
| **Preconditions** | User is logged in, property is viewable | |
| **Postconditions** | The property’s rating changes | |
| **Primary Actor** | User | |
| **Secondary Actors** | Rating database | |
| **Trigger** | User selecting a rating for the property | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User views a property |
|  | 2 | User selects a rating +1/-1 |
|  | 3 | Properties rating changes accordingly |
| **Extensions** | **Step** | **Branching Action** |
|  | 3a  **3b** | If the user selected +1 adds one to the properties rating  If the user selected -1 subtracts one to the properties rating |
| **Open Issues** |  | |

**2.4 Use Case 4**

|  |  |  |
| --- | --- | --- |
| **Number** | 4 | |
| Name | Login into user account | |
| **Summary** | The user logs into his/her account | |
| **Priority** | 5 | |
| **Preconditions** | User has created an account | |
| **Postconditions** | The user is logged into their account | |
| **Primary Actor** | User | |
| **Secondary Actors** | User database | |
| **Trigger** | The user clicks on the log in button | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User enters their username and password |
|  | 2 | The user’s information is verified |
|  | 3 | The user is logged in |
| **Extensions** | **Step** | **Branching Action** |
|  | 2a  2b | The user’s login information is incorrect  The user is returned to the login page |
| **Open Issues** |  | |

**2.5 Use Case 5**

|  |  |  |
| --- | --- | --- |
| **Number** | 5 | |
| Name | Renting a property | |
| **Summary** | Renting a property. The property should become “rented” by the user, and not be available for other users to rent | |
| **Priority** | 5 | |
| **Preconditions** | User has an account, the property is rentable | |
| **Postconditions** | The property will be rented by the user | |
| **Primary Actor** | User | |
| **Secondary Actors** | Google API,Property Database | |
| **Trigger** | Selecting the “Rent property” button | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User selects the “Rent property” button |
|  | 2 | The user will select when they want to rent the property |
|  | 3 | The property will be rented by that user till their time slot expires |
| **Extensions** | **Step** | **Branching Action** |
|  | **2a** | The user can select to cancel their renting if they change their mind |
| **Open Issues** |  | |

**2.6 Use Case 6**

|  |  |  |
| --- | --- | --- |
| **Number** | 6 | |
| Name | Select User Avatar | |
| **Summary** | User receives upload prompt to select an avatar | |
| **Priority** | 5 | |
| **Preconditions** | A registered user has logged in | |
| **Postconditions** | The image file that the user selected is uploaded to the database as his/her avatar and is displayed as such in references to the user’s avatar. | |
| **Primary Actor** | User | |
| **Secondary Actors** | User Database | |
| **Trigger** | Clicking change avatar button | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User clicks the change avatar button on the user page |
|  | 2 | Prompt opens for the user to upload a file to be used as an avatar |
|  | 3 | User selects image and it is cropped to fit in avatar pane |
| **Extensions** | **Step** | **Branching Action** |
|  | 3a | If the user selects an invalid file, the prompt will return with invalid file and restore the user’s previous avatar. |
| **Open Issues** |  | |

**2.7 Use Case 7**

|  |  |  |
| --- | --- | --- |
| **Number** | 7 | |
| Name | View User Page | |
| **Summary** | User views a personalized page showing rentals and settings | |
| **Priority** | 5 | |
| **Preconditions** | A registered user has logged in | |
| **Postconditions** | A personalized screen appears with user information such as username, properties rented before, etc… Also, personalization options are available at this screen. | |
| **Primary Actor** | User | |
| **Secondary Actors** | User Database | |
| **Trigger** | Clicking on the login button | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User logs in using User login previously defined |
|  | 2 | Window opens and displays the user’s profile |
|  | 3 | User can choose to update information in profile |
| **Extensions** | **Step** | **Branching Action** |
|  | 3a  3b  3c | If the user chooses to update password information a new screen will launch.  If the user chooses to update their avatar they will be prompted with a new action to do so.  If the user decides to log out then the user will be returned to the home screen. |
| **Open Issues** |  | |

**2.8 Use Case 8**

|  |  |  |
| --- | --- | --- |
| **Number** | 8 | |
| Name | Visit the Homepage | |
| **Summary** | View the landing page (index.html) | |
| **Priority** | 5 | |
| **Preconditions** | none | |
| **Postconditions** | The website’s homepage will display in browser | |
| **Primary Actor** | Any user | |
| **Secondary Actors** | Google API, django, mongodb, | |
| **Trigger** | Selecting the “Register property” button | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User types address of webpage into browser |
|  | 2 | User presses enter |
| **Extensions** | **Step** | **Branching Action** |
|  |  | None |
| **Open Issues** | Website is currently locally hosted | |

**2.9 Use Case 9**

|  |  |  |
| --- | --- | --- |
| **Number** | 9 | |
| Name | Register new user account | |
| **Summary** | The user creates an account with a password | |
| **Priority** | 5 | |
| **Preconditions** | None | |
| **Postconditions** | The User’s account has been created | |
| **Primary Actor** | User | |
| **Secondary Actors** | User database | |
| **Trigger** | The user clicks on the “Register” button | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User is directed to Register webpage |
|  | 2 | User enters their username and password |
|  | 3 | Database checks if user exists, adds user |
|  | 4 | Redirects to Home |
| **Extensions** | **Step** | **Branching Action** |
|  | 3a  3b | The user exists in the database  The user is returned to register page, error message displayed |
| **Open Issues** |  | |

**2.10 Use Case 10**

|  |  |  |
| --- | --- | --- |
| **Number** | 10 | |
| **Name** | Calendar | |
| **Summary** | Determines (using a displayable calendar feature) when a property is available for rent | |
| **Priority** | 3 | |
| **Preconditions** | User has an account and properties are uploaded and registered in the database | |
| **Postconditions** | The user will be able to view availability windows on the calendar | |
| **Primary Actor** | User | |
| **Secondary Actors** | Property Database, Calendar Object | |
| **Trigger** | Selecting dates on the viewable calendar | |
| **Main Scenario** | **Step** | **Action** |
|  | 1 | User selects the “Rent Property” button. |
|  | 2 | The user will select when they want to rent the property. |
|  | 3  4 | Click on the date range you wish to rent the property for.  The property will be rented by that user till their time slot expires |
| **Extensions** | **Step** | **Branching Action** |
|  | 2a | The user can select to cancel their renting if they change their mind |
| **Open Issues** |  | |

**3. Use Case Tests**

Use case testing is very important for product validation and verification. The final product should work as described. These tests explain how to test the requirements.

**3.1 Use Case 1 Test – Register Property**

Register a property with a name, location, description, and picture. Validate if the items added appear in the property database and are viewable to others.

**3.2 Use Case 2 Test – Click on Property Pin**

Click on a pin and see the image of the property, the property style (modern, cottage, etc), availability window, pricing, and a rating for the property.

**3.3 Use Case 3 Test – Rate Property**

Rate a property up or down and see that the rating registers within the ratings database.

**3.4 Use Case 4 Test – Login to User Account**

Log into an existing account. Make sure that the database verifies the account so that it can only be accessed with the correct password.

**3.5 Use Case 5 Test – Rent a Property**

Rent a property. Verify the database registers that the property is no longer available (rentable) by other users for the window of time that it was rented.

**3.6 Use Case 6 Test – Select a User Avatar**

Upload an image as an avatar for the user’s profile. Verify that the image has changed and is viewable to other users.

**3.7 Use Case 7 Test – Visit User Page**

Connect to a user’s page and view the information in their profile. Verify the information is correct in the database and is viewable in the application.

**3.8 Use Case 8 Test – Visit Homepage**

User types address of webpage into browser and presses enter to visit the landing page of

the application.

**3.9 Use Case 9 Test – Register New Account**

The user is directed to the register webpage where they can enter their credentials (username and password) to login to the application. Checking will take place with the database to see if the user already exists. If the user does not already exist, the user is added and redirected to the homepage.

**3.10 Use Case 10 Test – Calendar**

The user can select the date range on the calendar of when they would like to rent a given property.

**4. Non-Functional Requirements**

Below outlines criteria used to judge the system as a whole (priority 1 = lowest, 5 = highest).

|  |  |  |
| --- | --- | --- |
| # | Requirement | Priority |
| 1 | Uses the Google Maps API | 5 |
| 2 | Users can create accounts | 56 |
| 3 | Users can log in successfully | 5 |
| 4 | Properties can be rented | 4 |
| 5 | Users can put up properties for renting | 4 |
| 6 | Users can upload avatar images | 2 |
| 7 | System should use MVC architecture | 3 |
| 8 | Properties can be rated | 2 |
| 9 | Property availability windows can be viewed on a calendar | 2 |
| 10 | Code should be well documented | 2 |
| 11 | Application should be portable | 1 |
| 12 | System should use the Django web framework | 2 |
| 13 | The system implements a calendar feature | 2 |
| 14 | Application should be extensible and easy to maintain | 3 |

**5. User Interface**

See “User Interface Design Document” for more information about the UI design.

**6. Deliverables**

Deliverables include:

* Administrator Manual
* Systems Requirement Specification
* System Design Document
* User Interface Design Document
* Testing Report
* Code Inspection Report

**7. Open Issues**

None at this time.

**8. Appendix A – Agreement Between Customer and Contractor**

8. Customer Agreement The customer agrees to the capabilities of “Piece of Eden rentals” specified in the use cases and test cases outlined in this document. When and if changes occur, a new draft will be printed and signed by the customer and Team 3.

Client (print, signature)

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Team (print, signature)

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**9. Appendix B – Team Review Sign-off**

This document has been reviewed and agreed upon by all team members. Any disagreements or concerns are addressed in team comments below.

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**10. Appendix C – Document Contributions**

Rachel Cohen created this document with suggestions made from other team members.