CS3354 Software Engineering

Final Project Deliverable 2

Group 6

Project Title: Globe Hopper

Mohammed Ali-Khan

Minah Aqil

Edilberto Carrizales Cavazos

David Favela Corella

Shamsah Hossain

Henry Kim

Jacob Perez

Contents

Presentation Information

Our group has prerecorded our presentation:
https://drive.google.com/file/d/1OEWSztXoGdV5j3SCWL_Z5oHK8xTFwwX7/view?usp=sharing

Project Overview

Our group will be creating a virtual reality tourism app for a headset such as the Oculus or HTC Vive. Tourists will be able to experience various destinations in 360-degree views while tour guides will be able to record 360-degree footage (with or without commentary) and upload it. The app will include microtransactions for tourists to purchase VIP packages or virtual tickets to destinations. Customers will be able to find a destination using a search menu with filters by price, country, or biome. Tourists will also be able to subscribe to a tour guide or agency to be notified when they upload new experiences.

Instructor Feedback

> "A lovely and timely topic!! It is great that your group is trying to turn the covid times into an advantage. The tool you propose promises to facilitate travelling for potential tourists around the world.
>
> In the final report, please make sure to include comparison with similar applications -if any-, make sure that you differentiate your design from those, and explicitly specify how.
>
> Fair delegation of tasks.
>
> Please share this feedback with your group members.
>
> You are good to go. Have fun with the project and hope everyone enjoys the collaboration."

Addressing Feedback

In our second deliverable we will be sure to include a comparison to other applications that can be used to explore the world in virtual reality. We will be researching apps such as Google Tour Creator, and Seven Wonders.

GitHub Links and Task Delegation

Repository: https://github.com/6henrykim/3354-Team6

- Mohammed Ali-Khan
  - Github.com/darkSteal1152
  - Created the sequence diagram for purchasing a ticket or package
  - Created first GitHub repository – README file
  - Designed the test plans
  - Compiled Junit code for unit test on two modules
- Minah Aqil
  - Github.com/mxa190064
  - Created the use case diagram
  - Wrote the project conclusion
- Edilberto Carrizales Cavazos
  - Github.com/Eddie-Carrizales
  - Wrote the requirements specification
  - Estimated costs for hardware and software

- David Favela Corella
    - Github.com/DavidFavela
    - Chose the software process model
    - Designed the UI
    - Wrote the section on similar designs
- Shamsah Hossain
    - Github.com/ShamsahH
    - Created the sequence diagram for user login
    - Created the sequence diagram for searching for a tour
    - Created the sequence diagram for uploading a tour
- Henry Kim
    - Github.com/6henrykim
    - Created the sequence diagram for reading reviews
    - Designed the software architecture
    - Helped estimate costs for personnel
    - Edited the presentation video and added subtitles
- Jacob Perez
    - Github.com/Jacob1563
    - Created the class diagram

Deliverable 1

Software Process Model

The Globe Hopper project vision is to bring the world's best sights to people's homes. The way we could bring it to people is through the new VR technologies being developed today. VR is in its early days of being developed, as if being compared to cellphones or computers, only recently has VR been up for the public, which makes it hard to predict what the changes will be. For the Globe Hopper project to have a steady development and for us to make sure it works in the most well-known VR headsets, the software would need to be developed through the incremental development model.

The incremental development model allows for both an agile and structured plan for the development of our product. While we do have our minds set on what the final product should look like, customer feedback is of the utmost importance, as they will be the ones to experience our product. Using the incremental model also allows for a more feasible project, as undertaking the development of a new VR experience that works across multiple types of headsets and controllers is no easy task.

By using the incremental process model, prototyping our software would be possible. Using a prototype would also allow us to get community feedback from early adopters and give us an idea on how an open release of our product to the public would look like. Using this development model would allow for small incremental releases that will make sure our software is working consistently and will make continuous updates to ensure any bugs can be fixed.

One of the biggest reasons for the Globe Hopper project to use the incremental development model is due to a reason I mentioned before. VR is rapidly changing and expanding in the world. Taking extra time using a more safe but extraneous process such as the Waterfall model would only backfire due to time constraints and the different range of devices it would need to work on. The development model chosen allows for customer feedback through early releases of the product that can ensure it will hold up to people's standard.

Requirements

Functional Requirements

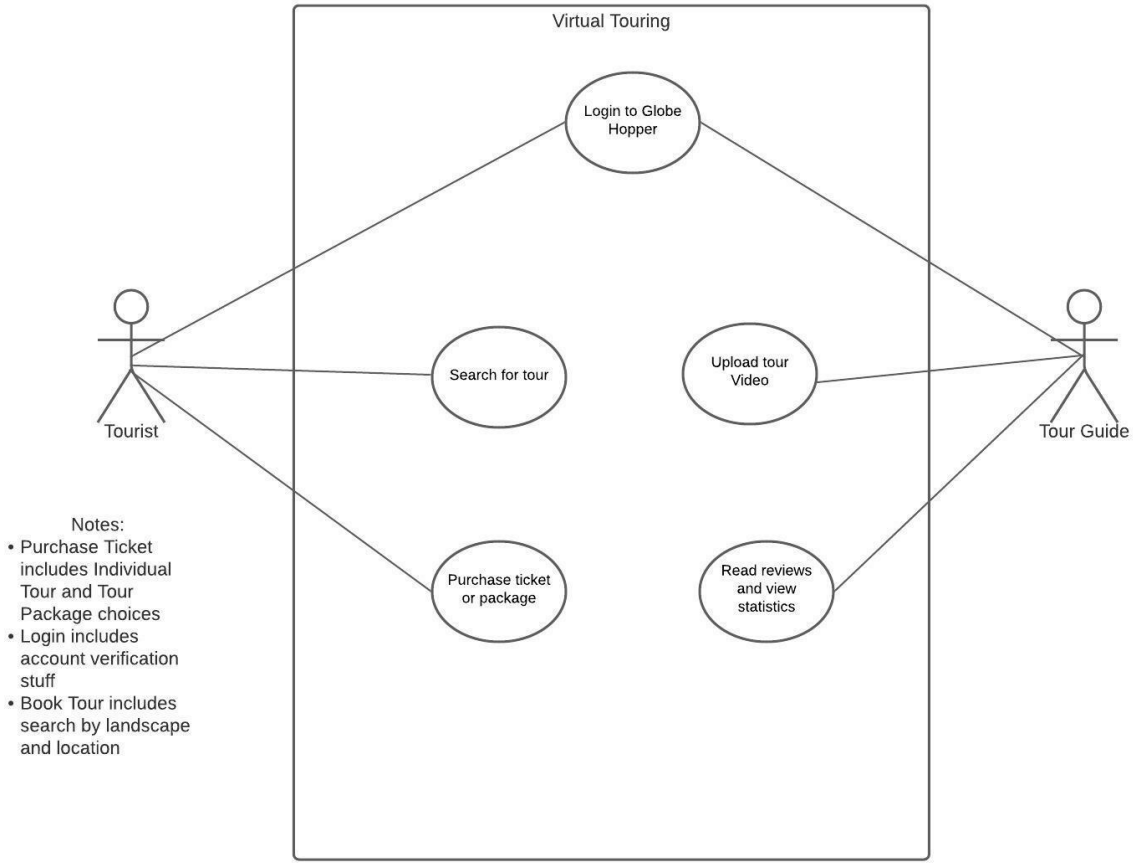
1. A tourist user shall be able to search for tour experiences by location or landscape.
2. A tourist user will be able to start tour videos, pause them, and exit them.
3. A tourist user will be able to buy tickets for a tour or purchase VIP packages from tour guides.
4. A tourist user will be able to subscribe to a tour guide or agency and will be notified when that guide or agency uploads new content.
5. A tour guide user or agency will be able to upload 360-degree footage and make it searchable by adding a description.

Non-Functional Requirements

1. Usability: The application is easy to learn, and a tutorial will pop up for the user. Control menus are also accessible from the application's settings to learn more or review application use.
2. Performance: The application has an overall medium usage of resources. Minimum processor required is a Qualcomm Snapdragon with 4 gigabytes of ram.
3. Space: The application takes up less than one gigabyte of storage space without any tours downloaded.
4. Dependability: The application depends on an active wi-fi connection of speeds greater than 50mbps per second. However, locations, landscapes, and other tours may be downloaded to access later (storage may vary by location).
5. Security: The Globe Hopper app does not share your personal information, and any information recorder (such as current location) is secured at all times by adequate encryption.
6. Environmental: The Globe Hopper app can be utilized in any location as long as there is enough physical space to walk around.
7. Operational: The application is online 24 hours a day 7 days a week including virtual tours. There will occasionally be updates to the system where the application be inoperable for a limited period.
8. Development: The Globe Hopper app will continue to add new tours and VIP packages each year.
9. Regulatory: All access or use of the Globe Hopper act by or for the United States Federal Government is subject to the "U.S Government Restricted Rights."
10. Ethical: The Globe Hopper Act takes the privacy of individuals seriously and believes in ethical correctness, for these reasons it will blur the faces, license plates, and any other sensitive information of any individuals that may appear in the tour videos, locations, or landscapes.

11. Accounting: The cost of utilizing the Google Earth API to utilize with the Globe Hopper app is $10,000 annually. Any additional implementations will cost an average of $70,000 dollars plus maintenance costs to add more tours and VIP packages which cost an average of $14,000 dollars per year. The cost for the user will mainly be the hardware which can range between $200 - $1000 dollars (minimum requirement can be met with $200 dollar hardware), plus the cost of the app and any additional VIP packages they wish to purchase (not required).
12. Safety/security: Due to the safety and security issues posed by access to satellite locations, The Globe Hopper app will only show locations of tourist attractions among other popular tours. Many of these locations are also not being recorded live and are thus a 360-degree collection of recent images.
13. Serviceability: While the Globe Hopper app does not require a lot of serviceability, it does occasionally require an update to locations that have changes as well as addition and upgrade of old tours.

Use Case Diagram

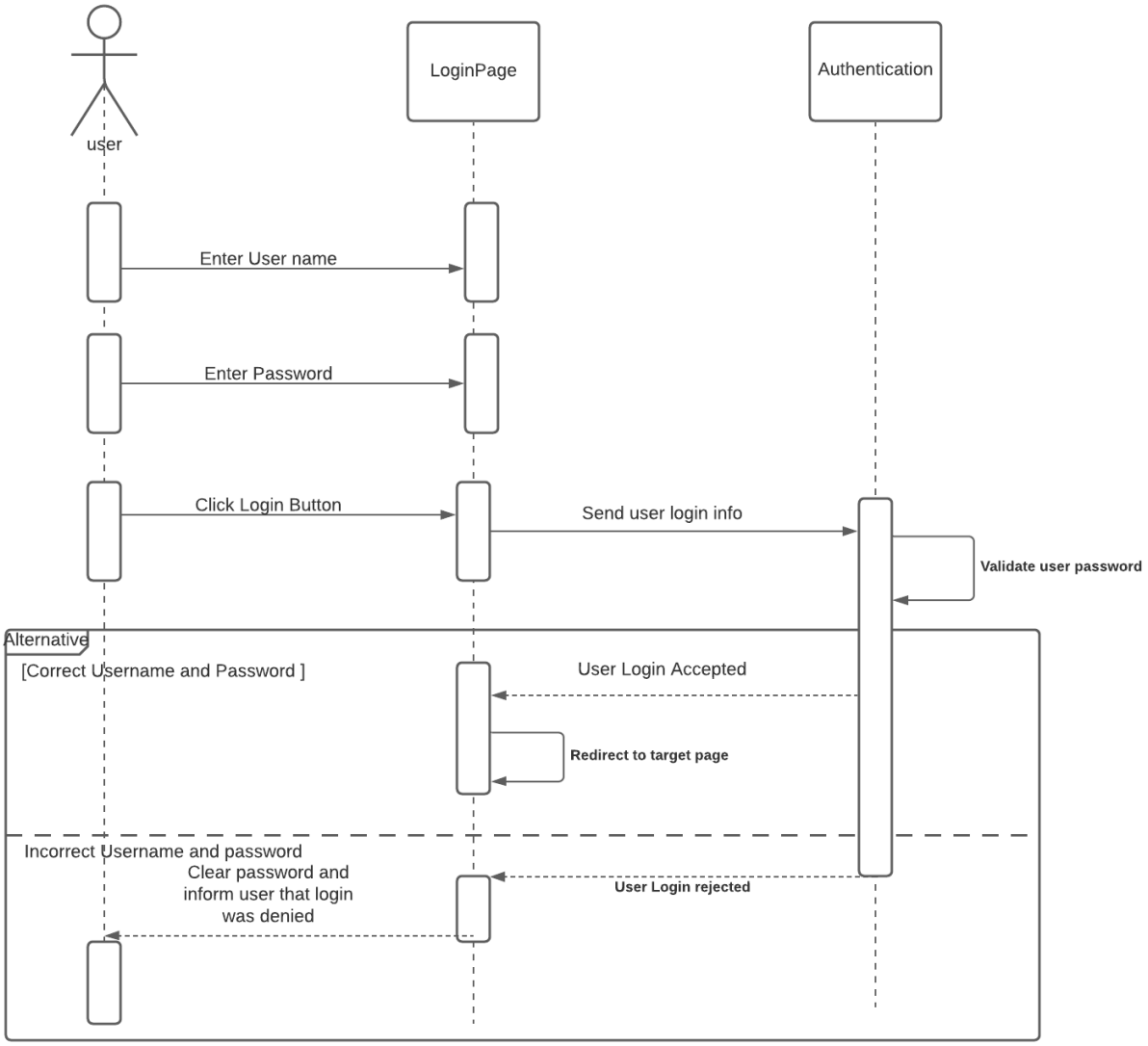


The use case diagram covers the basic functions and the scope of the Globe Hopper app. The actors are the tourist and the tour guide, and the use cases are for app login, which can be accessed by both the tourist and the tour guide. The search for tour and purchase tickets are the actions that are accessible to

the tourist, whereas the upload tour video and review and view statistics are actions accessible to the tour guide.

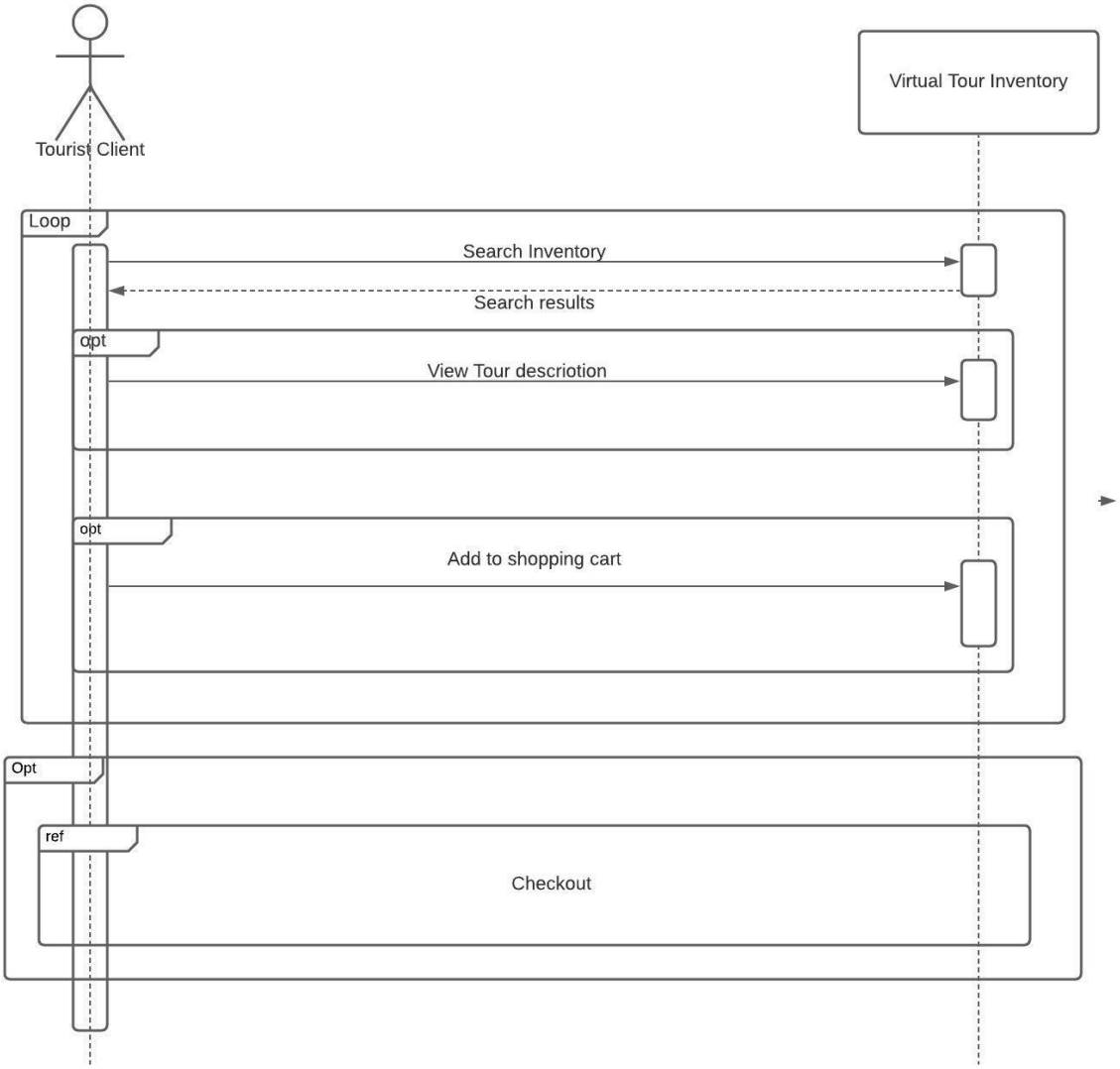
Sequence Diagrams

Login



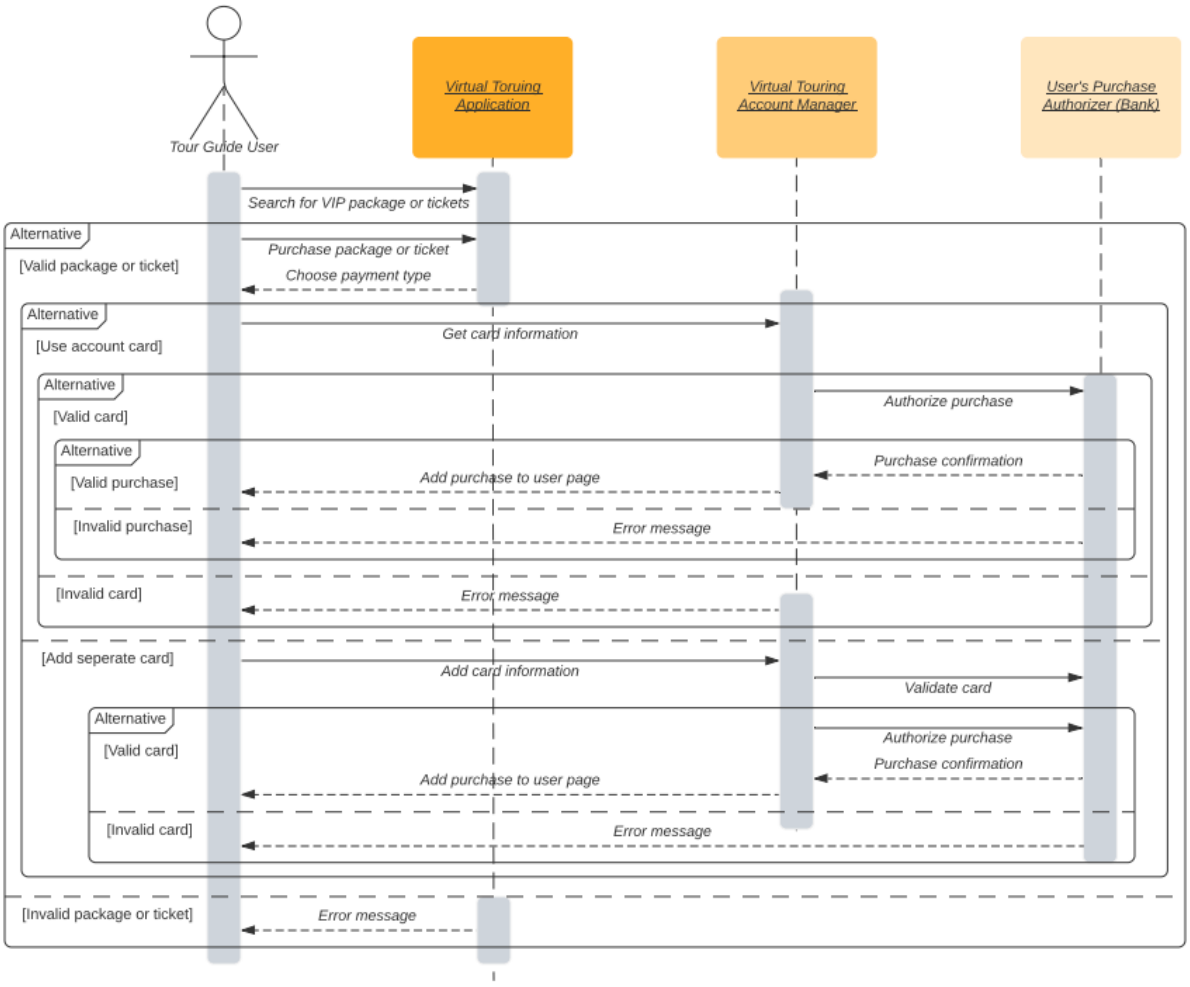
In order to use the Globe Hopper service, users will need to have an account that their purchases and uploads can be associated with. Users will enter their account credentials into the login page, and then their login attempt will be verified or rejected by an authentication service.

Search for tour



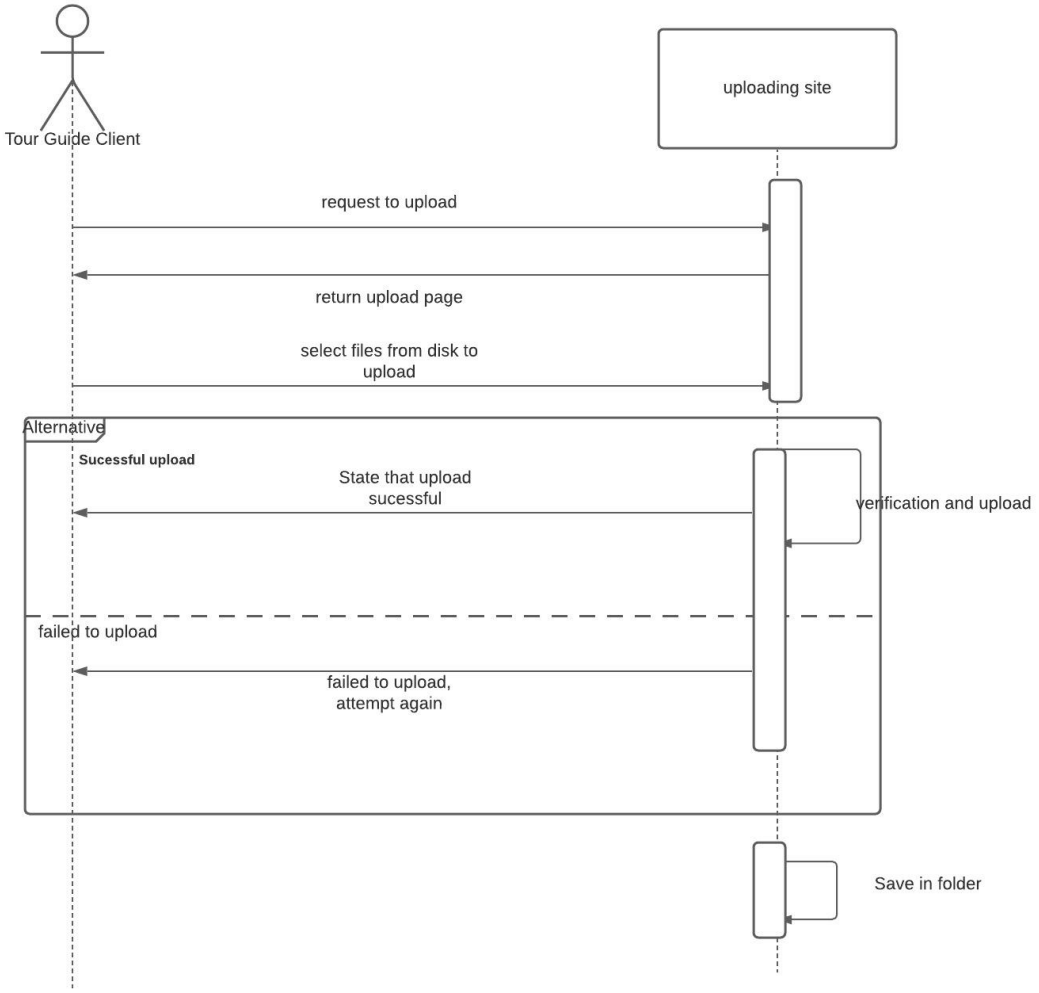
The tourist can search for tours in the virtual tour inventory and will be given results based on their criteria. The user can then look at that tour's description and make a decision on if they want to experience this tour or not. If the user decides to purchase the tour, they will add it to their shopping card and proceed to checkout.

Purchase ticket or package



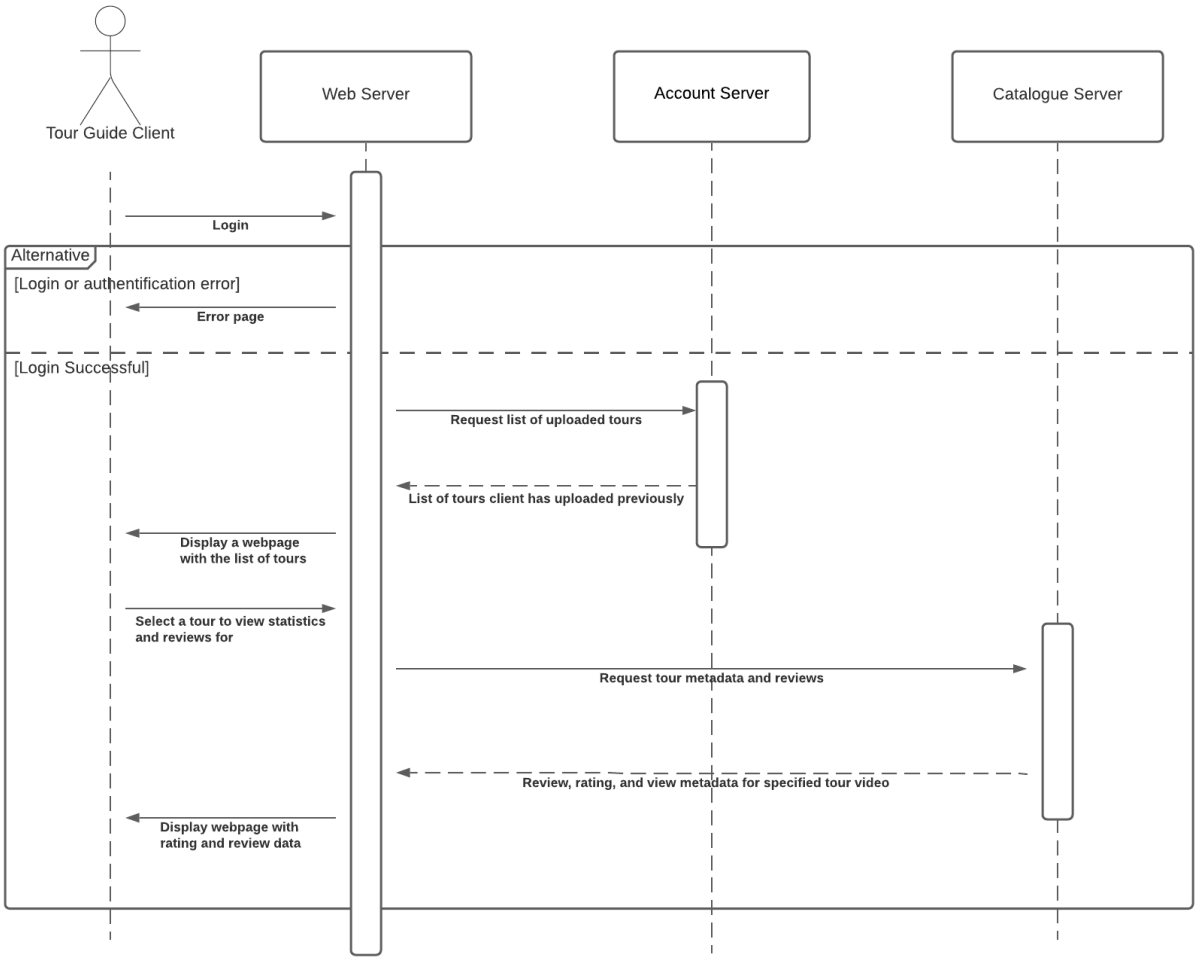
Special passes and VIP tickets can be purchased by the user separately through the app. The user can do this by having a valid card connected to their tourist account, or by using a separate payment method. The system will then verify, validate, and complete the transaction. If the payment method or card is invalid, the system will display an error message and the transaction will not be complete.

Upload tour



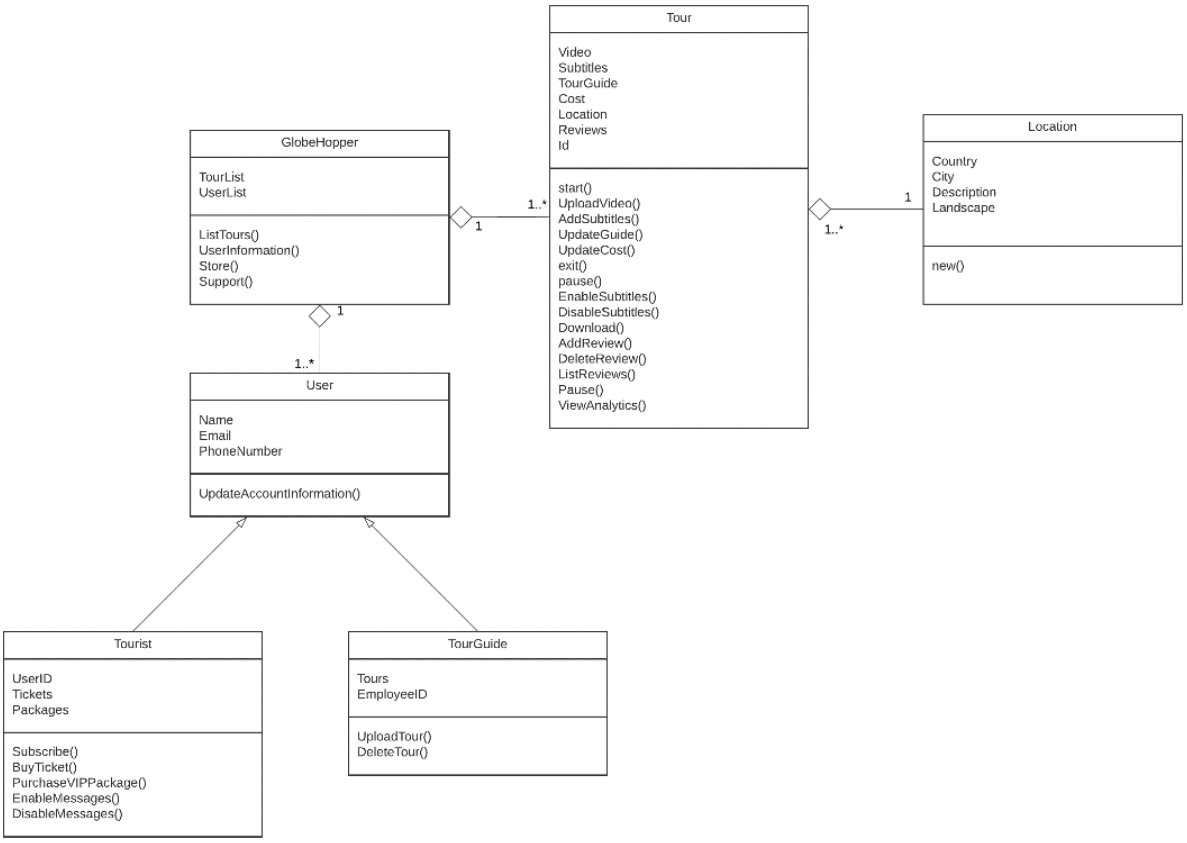
A tour guide can request to upload tour videos to the uploading site, to which the uploading site will redirect the tour guide to the upload tour page. The tour guide can then select the files from their disk memory and upload them. The tour video will be verified and if it is successfully uploaded, it will return a statement to the tour guide that the video was uploaded and save the video to the uploading site's folder. If the verification and upload failed, the tour guide will be notified that the upload failed and encourage them to attempt the submission again.

Read reviews and view statistics



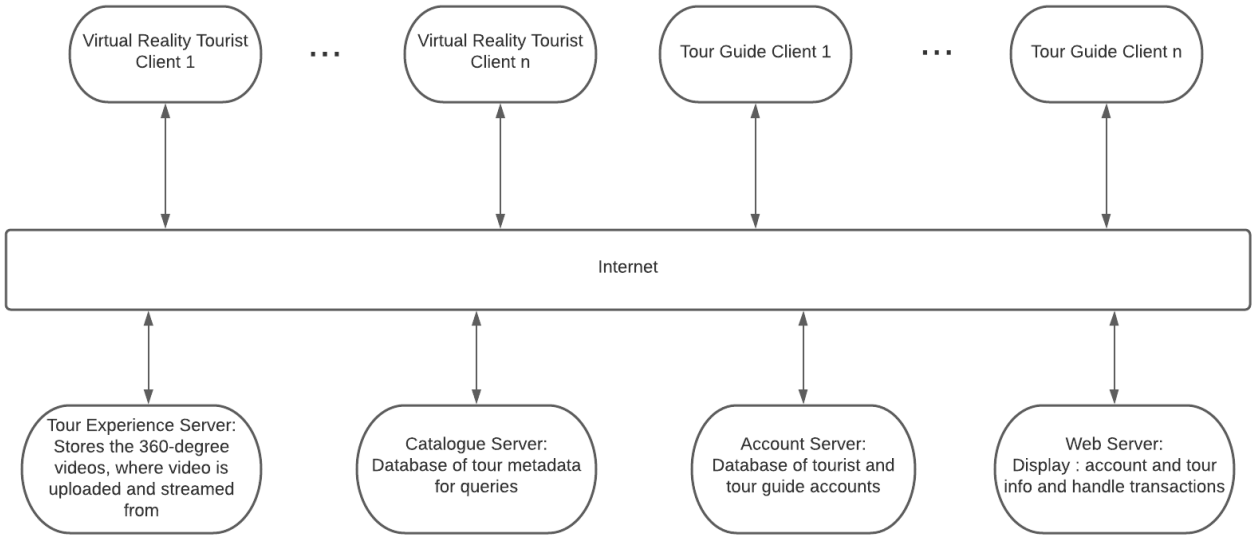
Once a tour has been uploaded, tour guides will likely want to read reviews and see statistics such as the number of views their tours are getting. Tour guides will do this by logging in to the web server which will display a list of tours they have uploaded to their account. The web server gets this list from the account server. After selecting a tour from this list, the web server will show the review data held in the catalogue server.

Class Diagram



The GlobeHopper application can have multiple users and multiple tours. Each user can be either a Tourist or a TourGuide. A Tourist will be able to subscribe to tours, buy either regular tickets for tours or VIP packages, and will be able to manage messages sent through the application. The TourGuides will be able to upload and delete tours, and will be provided with an employee ID. Each tour will have their own video, subtitles, TourGuide, cost, location, reviews, and ID. One tour can only have one Location, but multiple Tours can have the same Location.

Architectural Design



We have chosen to base our architectural design off the client-server pattern. This model is a good fit because it allows us to provide services over a large area and to multiple clients simultaneously. The tour experience server will store 360-degree videos and is where tourist clients will stream from. The catalogue server will contain a database of metadata on the tours such as tour agency, upload date, price, reviews, and locations. The account server will contain a database of account information for tourists and guides such as usernames and a list of videos purchased or uploaded. The web server will interface with the other servers and the client applications to display information to the users and let them upload, purchase, or view tour videos. The virtual reality tourist clients will be run on a VR headset such as the Oculus and the tour guide clients will be run from a desktop web browser. The client-server pattern is a particularly good fit for the incremental software process model we will follow since it makes the services modular, and it is easy to add new servers and features to it.

Deliverable 2

Project Scheduling

Start Date: January 1st, 2022

> We plan to start development on the first day of the new year. Although vaccines against COVID are available now, there are many who are still hesitant about traveling. Therefore, we would like to offer service soon.

End Date: January 28th, 2022

> According to our estimation with the function point algorithm, the first prototype should take about 2 weeks to complete. However, we do not plan on working during weekends and we plan to work 40 hours per week, so our schedule includes extra time to accommodate this as well as any employees needing time off for emergencies.

Cost, Effort, and Pricing Estimation

Function Point Estimation

	Function Category	Count	Complexity			Count x Complexity
			Simple	Average	Complex	
1	Number of user input	7	3	4	6	28
2	Number of user output	4	4	5	7	28
3	Number of user queries	4	3	4	6	16
4	Number of data files and relational tables	5	7	10	15	75
5	Number of external interfaces	6	5	7	10	42

GFP: 189

User input: 7

- Username

- Password
- Destination selection
- Review route option
- Video to upload
- Upload review/rating
- Payment information

User output: 4

- Downloaded tour
- Tour information
- Agency information
- Reviews

Number of user queries: 4

- Search for tour by destination
- Search for tour by location
- Search for tour by agency
- Search for tour by popularity

Data Files and Relational Tables: 5

- List of cities
- Table of costs
- Table of virtual tour length
- Table of most popular tours
- Table of user account information

External Interfaces: 6

- Headset - home page
- Headset - tour search and download page
- Headset - map page
- Desktop – tour upload page
- Desktop - account info page
- Desktop - settings page

0 (no influence), 1 (incidental), = 2 (moderate), = 3 (average), = 4 (significant), = 5 (essential)

- Does the system require reliable backup and recovery? = 5
- Are data communications required? = 5
- Are there distributed processing functions? = 3
- Is performance critical? = 5
- Will the system run in an existing, heavily utilized operational environment? = 2
- Does the system require online data entry? = 5

- Does the online data entry require the input transaction to be built over multiple screens or operations? = 3
- Are the master files updated online? = 4
- Are the inputs, outputs, files, or inquiries complex? = 4
- Is the internal processing complex? = 3
- Is the code designed to be reusable? = 2
- Are conversion and installation included in the design? = 1
- Is the system designed for multiple installations in different organizations? = 4
- Is the application designed to facilitate change and ease of use by the user? = 3

5 + 5 + 3 + 5 + 2 +5 + 3 + 4 + 4 + 3 + 2 +1 + 4 + 3 = 49

PCA = .65 + .01(49) = 1.14

FP = GFP * PCA = 189 * 1.14 = 215.46

Assuming productivity is 25 function points per person-week

E = 215.46 / 25 = 8.62 = about 9 person-weeks

Project duration:

Team size: 7

D = 9 / 7 = 1.29 = about 2 weeks

Estimated cost of Hardware Products

The cost for the user will mainly be the VR headset hardware which can range between $200 - $1,000 dollars [1]. We will need around $25,000 to buy computers for development and VR headsets for testing. We can start using Amazon Cloudfront as our video server for about $11 per month or $132 per year to service one region in the United States before we begin to expand the service [2].

Estimated cost of Software Products

The cost of using the Google Earth API is $10,000 annually. Additional implementations will cost an average of $70,000 dollars plus maintenance costs to add more tours and VIP packages which cost an average of $14,000 dollars per year

Estimated Cost of Personnel

Our team will start small with seven software engineers. The average salary of a software engineer is $117,000 [3]. We will also hire a database administrator who will be paid the average $95,000 salary [4]. To create tour content, we will hire 10 tour guides who will be paid the average salary of $33,000 [5].

Test Plan

For testing the software, a black box testing technique is employed, specifically Equivalence Partitioning. This test focuses on the functionality and behavior of certain modules, such as the user's username, password, destination selection, video query, given rating, and payment information. All these modules are a part of the software's input values, and a unit testing procedure can be used to determine whether a given input is valid and can continue to take action within the software, or is invalid, and will result in the appropriate error prompt so that the system as a whole is not negatively affected.

```
public class GHLogin {
    private String username;
    private String password;

    public GHLogin(String u, String p) {
        username = u;
        password = p;
    }

    public boolean startGlobeHopper(String user, String pass) {
        System.out.print("Username: " + user + " " + "Password: ");
        for (int i = 0; i < pass.length(); i++) {
            System.out.print("*");
        }
        System.out.println();
        if (user == this.username && pass == this.password) {
            System.out.println("Login Authenticated\nWelcome to GLOBE
HOPPER!\n");
            return true;
        } else {
            if (user != this.username) {
                System.out.println("Login failed - Incorrect username");
            }
            if (pass != this.password) {
                System.out.println("Login failed - Incorrect password");
            }
            System.out.println();
            return false;
        }
    }
}

import static org.junit.Assert.assertEquals;
import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.api.Test;

class GlobeHopperLoginTest {
```

```
    String user = "USERNAME";
    String pass = "pass123";
    GHLogin gh1 = new GHLogin(user, pass);
    @Test
    void test() {
        assertEquals("Here is the test for login authentication", true,
gh1.startGlobeHopper("USERNAME", "pass123"));
    }
    @Test
    void test2() {
        assertEquals("Here is the test for login authentication", true,
gh1.startGlobeHopper("FALSEusername", "pass123"));
    }
    @Test
    void test3() {
        assertEquals("Here is the test for login authentication", true,
gh1.startGlobeHopper("USERNAME", "password1234"));
    }
    @Test
    void test4() {
        assertEquals("Here is the test for login authentication", true,
gh1.startGlobeHopper("NOTUSERNAME", "pass456789"));
    }
}
```

For login authentication:

The unit test takes a given username and password and compares it to the predetermined username and password already present in the system. If all values are identical, login authentication is completed, and the user will be able to continue using the system. If the values are not identical or invalid, the login is not complete, and the user is not able to access the system.

```
public class GHRating {
    private int count;
    private double AVG;

    public GHRating() {
        this.count = 0;
        this.AVG = 0;
    }

    public boolean addRating(int newRating) {
        if (newRating < 0 || newRating > 5) {
```

```
            System.out.println("Rating must be between 0 and 5 stars\nCannot be "
+ newRating);
            return false;
        }
        if (count == 0) {
            this.count++;
            this.AVG = newRating;
        } else {
            this.count++;
            this.AVG = (AVG * (count - 1) + newRating) / (double)count;
        }
        System.out.println("Rating for this Tour is: " + this.AVG + "\n Last
rating given is: " + newRating + "\n");
        return true;
    }
}

import static org.junit.Assert.assertEquals;
import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.api.Test;

class GlobeHopperRatingTest {
    GHRating t1 = new GHRating();
    @Test
    void test4() {
        for (int i = 0; i < 5; i++) {
            int r = (int)(Math.random() * 5);
            assertEquals("Here is the test for rating authentication", true,
t1.addRating(r));
        }
    }
    @Test
    void test5() {
        for (int i = 0; i < 5; i++) {
            int r = (int)((Math.random() * 5) - 3);
            assertEquals("Here is the test for rating authentication", true,
t1.addRating(r));
        }
    }
    @Test
    void test6() {
        for (int i = 0; i < 5; i++) {
            int r = (int)((Math.random() * 5) + 3);
            assertEquals("Here is the test for rating authentication", true,
t1.addRating(r));
```

```
        }
    }
}
```

For rating validation:

The unit test ensures that only a rating between 0 and 5 can be inserted. Any other value will result in termination of the rating prompt. A valid rating value can then interact with the system, for example, contribute to the average of the tour video, or be displayed as the most recent rating.

Comparison to Similar Designs

Similar applications to ours have been made in the past. The most robust application until now is *Realities*, which allows users to experience "real-world locations into VR – interactive & with stunning photo-realism using photogrammetry" [6]. *Realities* only allow usage via the Oculus Rift and/or the HTC Vive. Content is uploaded by the developers, as they recreate the environments with computer-generated graphics, instead of using standard 360° footage. Their implementation allows the users to explore at their own pace the environments set by the developers. Content is regularly updated for free.

Google created their own implementation of virtual reality tours via the project *Tour Creator.* This app enabled users to generate their own 360° photos and upload them to Google servers [7]. Google also made this app compatible with most Android devices, desktop, and to be usable with Google cardboard. This app, which came with Google's Poly services that focused on virtual reality, was shut down along all Poly services on June 15th, 2021 [8].

Seven Wonders: Virtual Reality Tours is another app that shares features with ours. It enables users to see through virtual reality the seven wonders of the world [9]. The application does not need login information. It is implemented to work for Google Cardboard and takes advantage of the "gaze clicks" Google lets developers implement for users to interact with the content. The app is free.

Sites in VR is a virtual reality tours app made for iOS devices. No content can be uploaded from the users and no login is required to use it. The app contains a robust library of tours, allowing users to explore 15 countries in the world with a diverse number of tours per each country [10]. The app also supports different VR headsets and even allows users to customize their VR experience to better suit their viewing needs. The app is free.

Our app will differ from Seven Wonders and Sites in VR as we will emphasize partnerships with content creators to produce lots of high-quality content. With user accounts and ratings, we will also be able to track which types of tours are most popular and give users recommendations on the content they may be interested in.

Conclusion

Globe Hopper enables users to participate in virtual reality tours created by content creators with whom we collaborate. Travel has dropped dramatically as a result of the COVID-19 epidemic, which has limited where individuals can go. Virtual tourism uses technology, such as virtual reality, video, audio, narrative, static photographs, and other multimedia formats to provide viewers with an immersive experience of

an activity, location, or destination that they would not be able to acquire from photos or webpages alone. Viewers may see and experience a trip without having to travel there physically, which means they are not restricted by flight schedules, travel issues, safety concerns, etc. nor do they have to worry about weather conditions. They can do all this while saving a ton of money.

Our project itself did not require any changes, as we carried on with the original strategy. However, customer feedback and statistical reviews play a big part in our project and are extremely important to its future; we intend to make any necessary adjustments to accommodate the needs of our customers and keep our app up to date.

References

[1] M. Rakver," How Much Do VR Headsets Cost? Price Comparisons (2021)," smartglasseshub.com, [Online]. Available: https://smartglasseshub.com/vr-headset-prices/. [Accessed: Nov. 10, 2021].

[2] "AWS Pricing Calculator", *Calculator.aws*, 2021. [Online]. Available: https://calculator.aws/#/estimate. [Accessed: Nov. 10, 2021].

[3] "Software Engineer Salary | Salary.com", *Salary.com*, 2021. [Online]. Available: https://www.salary.com/research/salary/general/software-engineer-salary. [Accessed: Nov. 10, 2021].

[4] "Database Administrator Salary | Salary.com", *Salary.com*, 2021. [Online]. Available: https://www.salary.com/research/salary/listing/database-administrator-salary. [Accessed: Nov. 10, 2021].

[5] "Tour Guide Salary | Salary.com", *Salary.com*, 2021. [Online]. Available: https://www.salary.com/research/salary/posting/tour-guide-salary. [Accessed: Nov. 10, 2021].

[6] "Realities on Steam", *Store.steampowered.com*, 2021. [Online]. Available: https://store.steampowered.com/app/452710/Realities/. [Accessed: Nov. 10, 2021].

[7] "Tour Creator", *Tour Creator*, 2021. [Online]. Available: https://arvr.google.com/tourcreator/. [Accessed: Nov. 10, 2021].

[8] "Deprecation of Tour Builder - Google Earth Help", *Support.google.com*, 2021. [Online]. Available: https://support.google.com/earth/answer/10863849. [Accessed: Nov. 10, 2021].

[9] *Play.google.com*, 2021. [Online]. Available: https://play.google.com/store/apps/details?id=com.LAzyDevs.WondersVR&hl=en_US&gl=US. [Accessed: Nov. 10, 2021].

[10] "Sites in VR", *App Store*, 2021. [Online]. Available: https://apps.apple.com/tr/app/3d-mekanlar/id625987419#?platform=iphone. [Accessed: Nov. 10, 2021].