Group1



**Board of directors:**

Nazanin Binesh 300325667

Taban Nikdel 300337079

Rafael Olivares 300300098

**CEO**

Christy Kunjumon 300329467

Table of Contents

1. **Introduction 3**

History ……………………………………………………………………………………. 3

[What the Problem is 3](#_heading=h.3znysh7)-4

[Unified process methodology 4](#_heading=h.2et92p0)

Revenue Plan……………………………………………………………………………..4

2. **Analysis and Design 5**

[Functional Requirements 5](#_heading=h.tyjcwt)

[Non-Functional Requirements 5](#_heading=h.3dy6vkm)

[Use Case. 6](#_heading=h.1t3h5sf)

Dynamic Models (Sequence) 8

User interface Requirements (Sketches) 9

Design Goals 13

The architecture of the system 14

Introduction

# History

We are a group of four students who want to create a car rental web application with new features. Our group members are:Nazanin Binesh who graduated in IT and studying Computer at Douglas college, she has been working as a front-end developer and will play a programmer and analyser role in this company. Taban Nikdel who is studying computer at Douglas college as well is a programmer in this project. The next programmer is Rafael Olivares, who is a Douglas College student, he has a couple years’ experience working as a full stack developer. Christy Kunjumon who is pursuing computer at Douglas College and acting as CEO of Ranacharita, has experience working as Backend developer and Database Administrator.

# What the Problem is

Vancouver is one of the most beautiful cities in the heart of British Columbia, Canada. As a traveler or as a citizen, people want to explore this beautiful city. But sometimes the trips are too long to manage visiting everywhere. Using the public transportation or booking a Uber ride cost them a lot and they might miss their freedom and privacy. So, we introduce ***Ranachrita***, a car sharing web application. “Ranachrita offer customers to be their own boss.”  
If anyone got an extra car, we offer them to be a part of Ranachrita and earn some money and be part of this journey. On the other hand, we encourage people to register this web application and book cars for their long and short trips, in short- or long-term contracts. We offer our customers to be their own boss in their travels.

***Ranachrita* saves a lot of time.**

Using public transportation cause time consuming and expenditure, sometimes people get tired of stick in traffic or waiting for a long time to take a bus, whereas using private vehicle can save valuable time.

Renting a car would be a very helpful solution for people who do not own a car or want to use different cars.

beside driving in the city, customers can go to countryside or camping as we offer them a long road trip with ***Ranachrita*** rental car service.

***Ranachrita* provides more privacy.**

Using private vehicles are providing more privacy than using public transportation for people who do not have a car, especially in long road trips.

***Ranachrita* provides more safety.**

All cars are checked technically frequently in terms of customers' safety. They are checked up after every usage.

***Ranachrita* saves money.**

Renting a car is more reasonable than taking a taxi for every trip, even taking bus cause more expenses specially for travelling out of town. The car insurance, the costs of having a car and paying for the fuel will empty the owner’s pocket. It is simpler to just rent a car and enjoy trips.

# 

# In *Ranachrita* we use Unified process methodology

The software paradigm we are going to follow is the Unified process methodology.

It provides tools and criteria to help in software development procedures. It is widely used by developers in the field now a days and it is easy to adopt quickly. In addition, it comes with different managerial and technical complexity levels across various projects. Unified process methodology is known as iterative and incremental methodology. In this methodology, system develops through multiple iterations, with cyclic feedback and adaptation. The system is developed incrementally over time, iteration by iteration.

The Unified process methodology includes only 4 phases which are: Inception, Elaboration, Construction, and Transition. At the end of each iteration, the development team re-evaluates project priorities.

*Inception* –In this stage, we provide vision document, initial use case (Blueprint), providing initial business case model and we are preparing the project plan. Actually, this phase is a combination of planning and communication.

*Elaboration* – Is combination of modelling and planning so we are preparing our use case model, analysis model, architecture description and workflow.

*Construction* - In this stage we integrate the module also we do a test plan, test case, support documents and providing constructure manuals.

*Transition* – This stage is a combination of construction and deployment The team will deliver the completed project to the end users and help them learn how to use it. In this stage we will deliver a software that we try to release, based on the user feedback we are trying to add some features to the product and trying to implement these added options and send it for release step. After releasing, if any modification needed, we are going to this iteration one more time.

Like any methods, the workflow consists of: Requirements, Analysis, Design, Implementation, Test and Support that we are going through the above phases.

**Revenue Plan:**

Our revenue model is commission based where a user is charged a fee in any transaction.

Although, the cash collected by each journey in a trip is the only source of revenue for rental companies, Ranachrita’s revenue model is based on a variety of factors such as receiving commission from customer, host and also from the surge pricing and cancelation charges. Ranachrita takes around 15% of the fare charged to the customer, and the rest 85% is given to the host.

Trip Commissions:  
Ranachrita keeps 15% of each fare booked through its app, while its rival Turo takes 25%. The travelers pay around 45% less on average than a traditional rental company.

Surge Pricing: [Phase 2]

Ranachrita increases the rates based on certain algorithms. This helps Ranachrita to gain extra revenue and make better profits during peak hours and on special occasions.   
  
Cancellation fee:

As per Ranachrita’s policy, if anyone cancels the trip after booking it, they have to pay a cancellation fee.

Contract:

Ranachrita have the full authority to terminate or cancel the host account temporarily due to violation of policies.

Analysis and Design

# Functional Requirements

**Host:**

* Host should be able to create a profile
* Host should be able to define car(s) details (picture /technical information/)
* Host should be able to define a schedule for car(s) availability
* Host should be able see a list of requests for a car
* Host should be able to confirm who can rent his car.
* Host should be able to edit the details of the car
* Host should be able to define if insurance is mandatory
* Host should be able to define if adding more drivers are possible
* Host should be able to define Pickup & return location
* Host should be able to define if free delivery is the option

**Customer:**

* Customer should be able to Register a profile
* Customer should be able to list products
* Customer should be able to search and filter
* Customer should be able to look at details of cars
* Customer should be able to select a car with or without a driver
* Customer should be able to select insurance option
* Customer should be able to select having more than one driver in a trip
* Customer should be able to cancel a booked car
* Customer should be able to pay by card
* Customer should be able to select Pickup & return location
* Customer should be able to send feedback and rate

# Non-FunctionalzRequirements

* The Site should be able to serve a huge number of customers at the same time without problems.
* The site should be secure enough because there will be sensitive data.

# 

# Use Case

# 

# Dynamic Models (Sequence)

# User interface Requirements

# Customers can see all cars list and are able to filter the list. by clicking on one item the deatil of the car is shown and through this page they are able to book the car.

# List of cars click on Filter Icon

# D:\--Douglas-Summer2021\Software Engineering\Project1\ListOfCars.png D:\--Douglas-Summer2021\Software Engineering\Project1\Filter@1x.png



Car Detail Booking Policy

# D:\--Douglas-Summer2021\Software Engineering\Project1\Booking1@1x.pngD:\--Douglas-Summer2021\Software Engineering\Project1\CarDetail@1x.png

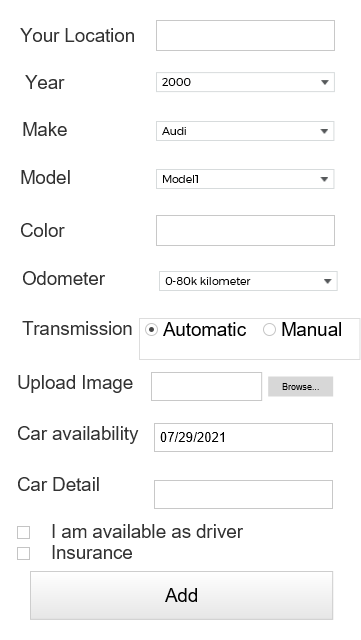
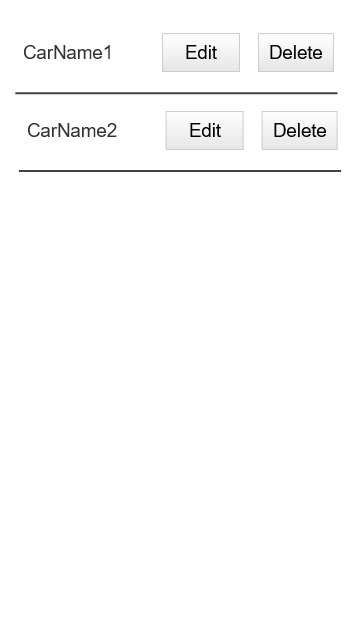
# 

# Message shown after confirm booking

# D:\--Douglas-Summer2021\Software Engineering\Project1\Booking2@1x.png

# Hosts are able to add cars to their profile and also they can see the list of their added cars, edit, and delete it.

# Host adds car List of added car by a host

# Design Goals.

### **Some Criteria for our Design Goals**

1. Performance Criteria

(a) Response time: For achieving the most reasonable response time we decided to use the light database like firebase DB and also implement the app and release it as an Android app.

2. Dependability Criteria

(a) Robustness: We are going to check all inputs that users can enter to prevent entering invalid data into the system.

(b) Fault tolerance: we are going to communicate with our users during app usage. meaningful errors must be shown to the user in faulty situations to alert them what should they do or what they have done wrong.

3. Maintenance criteria

(a) Extensibility: This system will be implemented modular and that is the reason it is extensible.

(b) Modifiability: Structured functional design helps us to modify functions easily.

(c) Adaptability: Android apps are the most popular among people all over the world as most of the people use android phones and it is easy to download.

(d) Portability: All people who have smart android phones are able to be our users and download the app.

4. End-user criteria

1. Usability: The software we design is very easy to use and understand. We design it as user-friendly as possible because we believe, users won't care what algorithms it uses, the UI is everything for the users. Our goal is to not bury functions under layers of meaningless icons and complicated pages. we present data in a straightforward way. we set screens up to match workflow, and size buttons according to their importance. During the design of the RANACHRITA App, we have thought about how to design this app to achieve the fulfill system according to the problems we had defined

# The architecture of the system

# RANACHRITA application will be implemented on the Android framework by Java language. Our Database is Firebase. The only needed infrastructure for the system to run is a smart mobile with Android OS. This Application is reachable after downloading and install it.

# A user registers through Register Component in the system and defines if they are Customer or Host. Also, there is another role called Admin.

# After registration, the user Login the system through Login Component, immediately after Login it will be checked on the database and determined that if the user is a host or customer to show them the relevant menu.

# Both customers and Host are able to edit their profile through Profile Component. They can also define Payment Methods through the Payment Component.

# The host creates a list of their car(s) in CreateCar Component, in this component the host is able to enter different data about the car. After adding a car the host is able to see the car(s) list through CarList Component and it is possible for them to edit or delete a created car.

# The host is also able to see a list of requests through RequstList Component. in this component they are able to accept or deny a request.

# The customer is able to see a list of the car through CarsListed Component. this is list is filterable by clicking on a filter Icon and Filter Component is run then.

# Customer can see the detail of a car by clicking on a cars block, the detail of the car will be shown through the CarDetail Component and in this component by clicking on booking button, Booking Component run and customer is able to book the car.

# Because the payment information is already defined by the customer it reduces the amount of money automatically.