CS673F14 Software Engineering
Group Project - UPark
Project Proposal and Planning

Your project Logo here if any

Team Member	Role(s)	Signature	Date
Fanghui Zhang	Team Leader		12/09/14
Xin Shan	Configuration Leader		12/09/14
Lu Zhang	Environment and Integration Leader		12/09/14
Rui Li	Requirement Leader		12/09/14
Xi Tang	Design Leader		12/09/14
Yilun Xie	Implementation Leader		12/09/14
Feiyu Shi	QA Leader		12/09/14
Mingli Li	Architect		19/09/14

Revision history

Version	Author	Date	Change
1.0	Yilun Xie		
1.1	Fanghui Zhang	11/07/14	

1. Overview

In Boston, people cannot easily find place to park their cars, especially in downtown area, or when some big events were held. We decided to build a online parking marketplace. We connect home and business owners who would like to earn money from renting their space with drivers in need of a convenient, safe and cost-effective place to park. The idea is simple just like our website. We manage the booking process and make sure users can find place to park.

2. Related Work

- **http://www.zipcar.com/**
  The Zip car has the same operation mode with our project, but we have a different function with it. The field of Zip Car is car rental, our project is about parking space rental.
- **http://ipark.com/**
  The iPark is a parking space rental service provider focuses on New York City, it has the same main function with our project, but we don’t provide payment function and we focus on rentals between individuals, also we plan to provide an online communication tool to

customers.

- **http://justpark.com/**
  justpark.com provides online services for drivers and owners, and it can guarantee that the parking you book will be avaliable on arrival, and whether drivers or property owner renting out a spot, they safeguard them against fraud and make sure online payments are protected.

3. Detailed Description

The project is divided into 6 parts, Account Management, Renter Interface, Public Park Place Interface, Tenant Interface, First-time User Instruction and Administrator Interface.

1. Account Management

Both renter and tenant should sign up or log in first. Tenant cannot be visitor because parking application should be possible to track.

1.1 Sign up

- Start with E-mail address or connect with Facebook
- If start with E-mail: Name, E-mail, password are needed

1.2 Password retrieve

- Need to enter Name + E-mail
- Then email a link to user’s email allowing to set a new password

1.3 Account setting

- Modify E-mail, Name, Password
- Enter phone number, home address, vehicle license number, whether is willing to rent out parking slot
- Renting Privacy: hide e-mail and phone number or not?
- Reminder: Whether to send an e-mail or text when a parking application is received, or his application has been approved?

1.4 Log in, Log out

- log in with facebook account or email + password

2. Renter Interface

After logging in, user can upload his own parking slots to rent out.

2.1 Create a Host

- Can you rent out your slot?
- Where? When? How much?
- How much credit should a tenant have?
- Description: including slot number, how to find it, payment methods, photoes

2.2 Modify Host

- Modify the slot information or description
- Delete the host

2.4 Tenant Evaluation

- Tenant evaluation process is generated automatically after a parking
- Rated by 5 stars. 0 is the worst, 5 is the best

3. Public Park Place Interface

At first, there will not be many renters, so tenant may find public park place from our web site. The places can be searched like renters. The Interface is similar to renter's.

- Can be searched by tenant, marked with PUBLIC
- Information: address, price, available slots, open periods, photo

4. Tenant Interface

After logging in, user can search parking places.

4.1 Search Interface

- Where do you want to park? (enter address)
- From when to when? (enter arrive time & depart time), or long time parking spot
- Other filters: acceptable price, distance from destination, public or private, renter credit

4.2 Search Result

- Displayed on Google map and a list below, sorted by distance by default
- Can be sorted by price, renter credit
- Renter information displayed: location, price, time, renter's name and credit
- Those unqualified searched result will be displayed below qualified ones. The unqualified information is highlighted.

4.3 Park Application

- Apply button
- When to park
- License number of vehicle going to park (his own vehicle by default)
- Can leave a message, for example, bargain
- Received message from renter can be viewed in a message box

4.4 Renter Evaluation

- Tenant evaluation process is generated automatically after a parking
- Rated by 5 stars. 0 is the worst, 5 is the best

5. First-time user Interface

First-time users are provided with instruction of how to use this product.

- Tutorial on the bottom
- Contact information
- FAQ

6. Administrator Interface

Administrator can collect information under privacy policy for product improvement and Sabotage control.

- Collect user's search record for user requirement research
- Review bad review user's credit and verify sabotage reports to prevent stealth sabotage
- Receive users' question to provide FAQ

4. Management Plan

4.1. Process Model

Scrum

- ➔ Scrum Roles
  - ◆ Product Owner: Fanghui Zhang
  - ◆ Scrum Development team: 8 members.
  - ◆ ScrumMaster: Xi Tang

- ➔ Scrum Meetings
  - ◆ Sprint Planning Meeting
    At the beginning of each iteration/sprint, the product owner is responsible for declaring which items are the most important. The team is responsible for selecting the amount of work they feel they can implement. The team pulls work from product Backlog to the sprint Backlog.

  - ◆ Sprint Review Meeting
    After Sprint execution, the team holds a sprint review meeting. As for this project, we feature a live demonstration in class.

The product owner declares which items I now consider done, the srummaster helps the prodcut owner convert the feedback to new product backlog items.

The sprint review meeting is also for Professor and all the classmates to atttend.

- ◆ Spring Retrospective Meeting
  Basically we did this on every Friday's meeting, before class and after class meeting.

➔ Scrum Artifacts

- ◆ Product Backlog
  We use pivotaltracker to implement the product backlog.
- ◆ Sprint Backlog
  Visible to the team Team. Team will discover additional tasks needed to meet the fixed scope commitment during exectuion.

4.2. Objectives and Priorities

The objectives of this project is to provide an information share platform for parking space information.

The priorities are information searching, information management, user privacy protection and online communication between customer and renter.

4.3. Risk Management

- Identifying Risks

Technical & Production

#Risk1: "Lack of skills and Knowledge about how to implement the google-map interfaces". All of our team members have no experience on developing webpage using google-map interfaces. This will lead to failure in implementing the search function of UPark project.

#Risk2: "Lack of skills and Knowledge about how to adequately use the j2e environment". Not all of our team members have enough experience on j2e.
This will lead to low quality of the UPark web server.

#Risk3: “Disunion opinions on how the website should be developed”. Since our project has 8 team members, each member has his own understanding on how the website should be. This may lead to great difficulty on final integration.

#Risk4: The team has some problems when using spring framwork.

Schedule & Managment
#Risk5: The process of the whole project is not very good due to bad project management skills.

Test
#Risk6: Product tests are not set up yet

Threat
#Risk7: The website may be vulnerable due to lack of security setting.

- Analyzing Risks
  Estimate the level of risk for idenfied and approved risks.
  #Risk1-Risk4: Low
  #Risk5: High
  #Risk6: Medium
  #Risk7: Hight

- Developing Risk Response Strategies
  - ➔ Acceptance: We will live with the risk that the team lacks of knowledge and skills, we can move on with learning the spring framework, learn to use github to do the version control, etc.
  - ➔ Avoidance: We designed some advanced functions for the website, but due to risk1-risk5, we might not implement these functions, we will in some term cut down unnecessary functions.
  - ➔ Control: Also called mitigation.Like risk7, we will try to minimize it, we will use different methods to make our website more secure.
  - ➔ Transfer: N/A

4.4. Monitoring and Controlling Mechanism

1. Monitoring Mechanism:
   - Track man-hour costs
   - Track Schedule
   - Track Scope

- Earned Values: We compare the value of work completed to date with the value of work supposed to be performed at that point in the schedule

2. Controlling Mechanism:
   If a change has occurred
   - Create change order & get approvals
   - Incorporate approved change into schedule
   - Measure the technical performance of a project and compare with the specifications required for project success.

4.5. Schedule and deadlines

1. Schedule
   Week 1 - Week 2:
   ...
   Week 8- Week 9: Finish sign in& sign up front end design & implementation, make spring framework work, and do the demo on week 9

2. Project Deadline
   12/01/2014

5. Quality Assurance Plan

(For more detail, please refer to SQAP document for encounter example)

5.1. Metrics

- **Produce Metrics**
  Size(KLOC), # of defects per KLOC
- **Process Metrics**
  Project effort (# of person hours), Productivity (hours per feature point). Working hours are kept in weekly reports. Productivity can be computed using points assigned to each feature in pivotal tracker and working hours on the implementation.

5.2. Standard

- **Meeting Time**
  On each Thursday, after class is over, a brief meeting is held. General design or implementation problems can be discussed.
- **Friday Coding Time**
  On each Friday, between 3pm and 6pm, members meet at 730 Commonwealth Ave CS Undergrad Lab. Code review and other reviews

can be conducted during the period. Other than that, members sit together and work together. It is good for helping each other out on problems and for the sake of efficiency.

- **Documentation**
  Documentation is very important in the whole process. It's a great treasure for each member. When something important is done, the member who did it are obliged to document what has been achieved and share it in the google drive folder. Apart from that, each leader should keep a document which records the work that has been done.
- **Coding Style**
  Please refer to the document "Coding style and more" which is continuously updated and the book "Clean Code". PMD and Checkstyle might be used to enforce it.

5.3. Inspection/Review Process

- **Code Review**
  During each Friday coding meeting, everyone who have submitted code by that day, should have his code reviewed by at least one other person. Reviewers are selected by invitation. Reviewer's responsibilities are: 1. finding obvious bugs 2. ensure quality of code and critique programming style. The total time of reviewing process should not exceed 30 min.
- **Story Review/Test**
  Once a feature story is finished, it is delivered to the design or implementation leader on pivotal tracker. Leaders shall ensure the correctness and integrity of the feature story before it goes to release.
- **Iteration Review/Test**
  Before each iteration ends, a new version shall be released. Before the release, all the finished features should be reviewed by the project leader, design leader, implementation leader and QA leader. A client representative may be invited. This could be done during Friday coding meeting.

5.4. Testing

- **Unit Test**
  This is performed by each individual developer when he finishes one piece of code. "JUnit" is one choice.
- **Integration Test**
  Integration test will be conducted by implementation leader and integration leader before each module is fully tested.
- **Acceptance Test**
  Acceptance Test is conducted before each release by client representative and project leader, or could be done together with iteration review.

5.5. Defect Management

(e.g. describe the criteria of defect, also in terms of severity, extend, priority, etc. The tool used to management defect, actions or personnel for defect management)

- **Criteria**
  Any deviation from the requirement or design are counted as a defect.
- **Severity**
  Major, trivial. major defects must be treated right away.
- **Priority**
  High, low. major defects have high priority.
- **Management**
  When a defect is discovered by customer or developer, a bug story is proposed on pivotal tracker. The project leader will see to it and assign it to the most relevant developer(s). The assignee should attend to it as a top priority and resolve it as soon as possible. Inform the project leader when it's done.

6. Configuration Management Plan

(For more detail, please refer to SCMP document for encounter example)

6.1. Configuration items and tools

- Version control tool: git
- Project repository on Github
  https://github.com/CS673-Project/UPark
- Issue (bug) control tool: Github issues
  https://github.com/CS673-Project/UPark/issues
- Deployment tool: Capistrano

6.2. Change management and branch management

- Everyone should have their own branch registered with his own name.
- All deliverables are on “master” branch.
- Merge into “master” branch should be only performed by deployment owner.
- Feature/bug branches will not be existed before first prototype.

6.3. Code commit guidelines

- Pull “master” branch before starting a day’s work.
- Commit constantly to local.
- Push to remote to wrap up working.
- Deploy via Capistrano script.
- Deployment owner will do weekly deployment on Friday, 5:30 pm.

7. References

(For more detail, please refer to encounter example in the book or the software version of the documents posted on blackboard.)

8. Glossary