Big Blue Parking Genie

Project Overview

This project aims to build a web application for managing and selling parking for specific time slots.

The system will allow a user to rent out a parking spot, specifying the specific time slot that they want to rent it. The user will be given a form of identification that can be given to allow entry to the parking zone. The system will also have a manager mode that will allow for seeing how much space is left, dealing with discrepancies, handling customer complaints, and overriding errors in the system, etc.

Team Organization

Anne Jetton - Project manager, responsible for time management and due dates for specific elements, along with over all work distribution, as well as contributions to code.

Kate Sargent - In charge of overall CSS and html development, Checking for correctness in CSS code, design input, etc.

Makalee Beelek - In charge of overall Python development, Checking for correctness in python code, design input, etc.

Derek Ward - In charge of overall Django development, Checking for correctness in the django framework and organization, etc.

This team intends to stick to their specific roles first, then lend a hand in the other departments where time and skill allows. We also intend to check each other's work for correctness, as well as help each other to devise unit tests to ensure absolute functionality of the application.

Software Development Process

The development will be broken up into five phases. Each phase will be a little like a Sprint in an Agile method and a little like an iteration in a Spiral process. Specifically, each phase will be like a Sprint, in that work to be done will be organized into small tasks, placed into a “backlog”, and prioritized. Then, using on time-box scheduling, the team will decide which tasks the phase (Sprint) will address. The team will use a Scrum Board to keep track of tasks in the backlog, those that will be part of the current Sprint, those in progress, and those that are done.

Each phase will also be a little like an iteration in a Spiral process, in that each phase will include some risk analysis and that any development activity (requirements capture, analysis, design, implementation, etc.) can be done during any phase. Early phases will focus on understanding (requirements capture and analysis) and subsequent phases will focus on design and implementation. Each phase will include a retrospective.

	Iteration
	Phase 1 - Requirements Capture
	Phase 2 - Analysis, Architectural, UI, and DB Design
	Phase 3 - Implementation, and Unit Testing
	Phase 4 - More Implementation and Testing

We will use Unified Modeling Language (UML) to document user goals, structural concepts, component interactions, and behaviors.

Communication policies, procedures, and tools

Our team will be meeting every saturday, from noon to 2pm, to discuss and look over last week's assigned work, to check for correctness and completeness, and fix any bugs, as well as check to see if the individual assigned pieces will work together well. We will also discuss what needs to be completed the coming week, and assign work accordingly. Work will be pushed to the repo after each of these meetings. We will be meeting via discord, and zoom where appropriate.

Risk Analysis

1. Customer Can Reserve A Parking Spot
    1.1. Likelihood - Medium
    1.2. Severity - High

1.3. Consequences - We will be unable to create any part of our application except the customer and manager accounts. Essentially the application would be useless.
1.4. Work Around - The University would continue to manage parking as they have been.
1.4.1. Difficulty - Easy
1.4.2. Impact - The University would not save any more time than they already do.
1.4.3. Pros - The University already has a system for parking which is familiar to them.
1.4.4. Cons - We would not have learned how to work as a Software Development Team. We would not have a functioning application. We would fail this class.

2. Customer Can Cancel A Reserved Parking Spot
2.1. Likelihood - Medium
2.2. Severity - Medium
2.3. Consequences - None
2.4. Work Around - The user could email the parking manager to manually refund them
2.4.1. Difficulty - Easy
2.4.2. Impact - The parking manager would have to spend more time dealing with angry customers
2.4.3. Pros - Less work for the development team.
2.4.4. Cons - The process would not be automated.

3. Parking Spots Are Not All The Same (Price, Distance, etc.)
3.1. Likelihood - Medium
3.2. Severity - Medium
3.3. Consequences - We would not be able to implement the maps feature.
3.4. Work Around - Have the parking attendant tell each person where their actual parking spot is and take their money
3.4.1. Difficulty - Low
3.4.2. Impact - The parking attendant would spend more time dealing with customers. The customers would spend time driving to get to their parking spot.
3.4.3. Pros - Less work for the development team.
3.4.4. Cons - The process would not be as automated.

4. Customer Must Create A User Account
4.1. Likelihood - High
4.2. Severity - High
4.3. Consequences - The entire application would not be able to purchase or cancel a parking spot
4.4. Work Arounds - The parking attendant would take their money for a parking spot.
4.4.1. Difficulty - Low

        4.4.2. Impact - The parking attendant would spend more time dealing with customers.
        4.4.3. Pros - Less work for the development team
        4.4.4. Cons - The process would not be automated.
5. Managers Must Create A User Account
    5.1. Likelihood - High
    5.2. Severity - High
    5.3. Consequences - The parking spots would not be able to be managed individually.
    5.4. Work Arounds - The development team would update the site to match the availability of parking spots and create events.
        5.4.1. Difficulty - High
        5.4.2. Impact - The development team would spend a lot of time constantly updating the application
        5.4.3. Pros - Parking Lot Managers would have less work.
        5.4.4. Cons - The development team would spend too much time.
6. A Lot Attendant Can Verify A Customer's Purchase
    6.1. Likelihood - High
    6.2. Severity - Low
    6.3. Consequences - None
    6.4. Work Around - No customer verification is required
        6.4.1. Difficulty - Low
        6.4.2. Impact - Parking Lot Attendants would not have a job.
        6.4.3. Pros - The development team would not have to create individual QR codes. The University would save money because they would not have to hire Parking Lot Attendants.
        6.4.4. Cons - No one would purchase parking spots because they would not be verified anyways. Our application would not be used.
7. Parking Spots Are Managed Individually
    7.1. Likelihood - Medium
    7.2. Severity - Medium
    7.3. Consequences - None
    7.4. Work Around - Have the parking attendant tell each person where their actual parking spot is and take their money
        7.4.1. Difficulty - Low
        7.4.2. Impact - The parking attendant would spend more time dealing with customers. The customers would spend time driving to get to their parking spot.
        7.4.3. Pros - Less work for the development team.
        7.4.4. Cons - The process would not be as automated.
8. Application Is Browser Based And Mobile Optimized
    8.1. Likelihood - High
    8.2. Severity - High
    8.3. Consequences - None
    8.4. Work Around - Require users to download the application on their computer.

        8.4.1. Difficulty - Medium
        8.4.2. Impact - More time spent for users and lot managers to download and constantly update the application.
        8.4.3. Pros - Developers save time that would be spent creating a user friendly web based application.
        8.4.4. Cons - Customers and Managers would not use the application as often due it not being as user friendly.
9. The Development Team Will Meet Weekly
    9.1. Likelihood - High
    9.2. Severity - High
    9.3. Consequences - We will not be able to implement any features
    9.4. Work Around - We could meet less often or not at all.
        9.4.1. Difficulty - Low
        9.4.2. Impact - More time would be needed from each member to individually update the progress on github.
        9.4.3. Pros - The development team would not have to socialize.
        9.4.4. Cons - The application would take more time and be less polished.
10. Application Can Provide Discounts
    10.1. Likelihood - Medium
    10.2. Severity - Low
    10.3. Consequences - None
    10.4. Work Around - Parking Lot attendants can refund users if there is a discount at the parking lot
        10.4.1. Difficulty - Low
        10.4.2. Impact - More time would be needed by parking lot attendants to refund each customer with a discount. Parking lot managers would need to be sure that attendants have cash or card readers to be able to refund customers
        10.4.3. Pros - Less time spent for the development team.
        10.4.4. Cons - More work for the parking attendants and managers
11. Application Will Show On Google Maps Where The Parking Spot Is Located
    11.1. Likelihood - Low
    11.2. Severity - Low
    11.3. Consequences - None
    11.4. Work Around - Give the User an address to the parking lot
        11.4.1. Difficulty - Low
        11.4.2. Impact - More time would be spent by the user to find the spot using a separate map application.
        11.4.3. Pros - Less time spent for the development team.
        11.4.4. Cons - Users would need to spend more time finding their parking spot.
12. A Countdown Clock Will Show How Much Time Is Left Until The Event
    12.1. Likelihood - High
    12.2. Severity - Low
    12.3. Consequences - None

12.4. Work Around - Show the date and time of the event.
    12.4.1. Difficulty - Low
    12.4.2. Impact - Users would spend more time calculating how much time they have left until the event.
    12.4.3. Pros - Less time spent for the development team.
    12.4.4. Cons - Users would have to calculate how much time they have left themselves.

Configuration Management

See the README.md in the Git repository.