Parking App

Project Overview

This project aims to create a UI for individuals seeking to sell parking and find parking at local events.

The parking app will be for Logan-based parking. Users will be able to either register their parking spaces, or find and reserve parking spaces from the comfort of their home. This can reduce traffic before local events, such as USU basketball and football games. It also provides a unified marketplace where a user can find any available parking, instead of browsing multiple websites/locations to find their parking.

Team Organization

Web Server: Responsible for serving up the JS app to users through the web browser. Team member Max Thomas will work on this portion of the project

Web App: Responsible for allowing Users to reserve parking spots and for lot owners to register, share and modify lot info. Team member Ethan Bailey will work on this portion of the project

Backend Application: Responsible for tracking accounts, lots and managing parking spot availability. Team members Kaleb Dutson and Brandon Jolley will work together on this portion of the project

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.

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

Communication Policies:

- All communication will be done through the team's Discord server.
- Manage group work and tasks through Trello. Communicate with group members through Discord to assign tasks to group members.

Procedures:

- **GitHub Repo Procedures:**
  - The **master** branch is for finalized code only. Push your testing code to a separate branch.
  - The code will be modular, so each branch will be thoroughly tested before being added to the main project.
  - The **dev** branch contains code to be tested before being pushed to the **master** branch. This will prevent breaking code in **master**.
  - Notify the group via Discord once you have pushed new, working code to the **master** branch.
- **Weekly Meetings:**
  - At the beginning of the week, we will discuss any major additions/subtractions we will be making to the project.
  - We will identify possible areas of conflict and brainstorm potential solutions
  - We will assign a task to each team member to break any problems down into smaller, more manageable portions.

Tools:

- Django
- GitHub: code repository
- Google Docs: Collaboration on non-code files.
- Discord: Group communication.
- Draw.io: Creating diagrams and UML.
- Trello: Managing project's tasks.
- Python: server
- Javascript: Web App
- Java: Parking Lot Application (Backend)
- MySQL: Database

Risk Analysis

Trello Project Organizer:

- Any issues that team members run into during the week will be posted on the Project's Trello. This will prevent any team member from accidentally interacting with broken or faulty code.
- We will utilize Trello's tags to keep track of any high risk features.

QR code:

- **Likelihood:** No one in the group has ever worked with an app that uses QR codes. There is a moderate likelihood that we will not implement the QR code feature.
- **Severity:** Mild
- **Consequences:** The app would just need an alternate method of authentication for users arriving at their reservations. An alternative would be sending a verification code to the user via the app, email, or text.

MySQL:

- **Likelihood:** The team is familiar with databases and figuring out how to connect data to the backend of the app shouldn't be too difficult. Low likelihood of not implementing a MySQL database.
- **Severity:** High
- **Consequences:** The database is crucial to the functionality of the app. If we can't use MySQL, we will have to use an alternative database.

Configuration Management

See the README.md in the Git repository.