CS Senior Design Specification report

Parking App

Kavleen Kaur & Jumana Rahman

Mr. Thomas Re

Fall 2021

December 2, 2021

Chapter 1 Project Description and Goals

Section 1.1: Project Scope and Vision

In the 21st century, driving is one of the most popular forms of transportation faculty and college students use to get to school. However, in order to assure that they arrive in time for their first class for the day, faculty and students must first account for traffic they may encounter along with the time it may take them to find a parking spot on campus. This project entails building a web application that will provide a solution for students and faculty that struggle finding parking spots on campus. This application will allow students and faculty to save time when looking for parking by allowing users to see if there are any available parking spots on Hofstra's North and South Campus parking lots. Once students and faculty arrive on campus, instead of driving around looking for parking, they will be able to check the availability of parking lots near their first class and depending on that, users will either park their vehicle or look at another parking lot to see if there are vacant parking spaces there. After parking, students and faculty will check-in to where they parked their vehicle and will check-out when they leave the parking spot vacant. The web application will update whenever a user checks-in to show that the spot is taken and will do the same when a user checks-out to show that the spot is now available.

This application will also assist the university carry out procedures like construction or events. For example, in the scenario where the university needs to have a specific parking area completely vacant, Public Safety would usually look up and contact the owners of the vehicles parked there. This project's parking application will allow users that have been granted a user status of admin to find a vehicle's owner easily by clicking on occupied parking spaces at a parking lot and seeing the pop up of the vehicle owner's information like their name,license plate number, vehicle's year, vehicle's make and model, phone number, and email.

Several parking apps have been made for the general public to use in order to reserve parking spots for an event planned. For example, the BestParking app allows users to find and reserve parking for a small cost. It is stated on their website that users can "book a space in just a few easy clicks...[4]". This isn't ideal for a university because it would simply lead to a competition amongst students and faculty on who can reserve the parking spots closest to the building they will be having class. Another parking app that serves universities is ParkMobile. Using ParkMobile students and faculty are easily able to find and pay for a parking spot and use a timer to show how much time the user has left for their spot so that they can pay on the app to add more time for their parking spot. However it is stated on ParkMobile's user's frequently asked questions webpage that, "ParkMobile Pro is our premium membership service. For $0.99/month, you receive additional features and benefits like real-time parking availability, discounts on parking transaction fees, roadside assistance, and more [2]." This reveals that in order to have access to real-time parking availability on the app, users must have a membership in ParkMobile Pro. In addition to that, East Tennessee State University made an app called ETSU that has a

feature that enables users to find parking at their university for free. It is stated that ETSU's system, "counts cars as they enter and leave parking lots and garages and then feeds that data to a 'field hardware device' that uploads it to cloud storage [3]" to determine if there are parking spots available for students at a specific parking lot. Unfortunately ESTU is only available for East Tennessee State University and hasn't been out for other universities like Hofstra to use. This project will provide a more precise location for the available parking spots than ETSU because instead of showing a number of the available parking spots left, the web application will show the exact locations of the vacant parking areas. Unlike most of the parking apps that exist today, the target audience this project adheres to is a university that provides free parking for students and faculty. Our app provides real time parking availability with no cost along with gives the university's parking management system access to all the information on the vehicle's car owners in every parking lot.

The prototype's main features will be account registration, secure login, location sensor/sharing, database connection, and interactive user interface showing grid maps of the various parking lots on Hofstra campus that you can zoom into or allow location sharing to find which lot you are closest to. Other features include user vehicle reporting system and estimated check-out time when successfully checked-in.

Account Registration and secure login will allow users to register as a student, faculty, admin, or guest. Guests have the same access as students. Users can login and check-in a parking spot after they have parked their vehicle and check-out when they make the parking spot vacant again.

The grid maps will have a color coordination and numbering system that will number all the lots by letters and the individual parking spots by numbers (ex: A1). The color coordination system will have all spots that are taken gray, open spots will be clear, and all spots not available to the current user type will be red. Users can check-in by clicking on the spot on the grid map. While checking-in, the user will be prompted to input an estimated check-out time. After successful check-in, the parking spot will be gray and cannot be clicked on until successful check-out.

User interface will use HTML, CSS, and JavaScript. What is shown and done in the app is different for students, faculty, and admin. For example, students cannot check-in a faculty-designated parking spot, however, a faculty member can check-in to all available parking spots. Another example is a student or a faculty member cannot access the information behind a parking spot, i.e. vehicle owner and contact information, however, an admin such as Public Safety can. The user interface will be responsive for all users with different phone sizes and types to easily see maps and parking spots.

Location sharing will be an important feature to counter users reserving spots before reaching campus or before leaving their dorm room. This will give all users an equal chance to find parking spots in an efficient manner. Users can view open parking spots but cannot check-in until they reach it.

An extra feature to add into this prototype in the future is having a machine sensor that communicates with the app to see if a parking spot is taken. The machine sensor would be able to read the license plate and log the vehicle's owner into the system. Also in the future, the parking web app can be linked to a hofstra student account.

This web application differs from current solutions because it is a more personalized approach to a university's parking system. It allows users to see an updated and presentable version of the current status of the parking lots. Existing solutions like parkhub are generalized approaches to fit all parking accommodations for events such as stadium and concert parking. In this case, parking should be free and accessible for all hofstra persons. This prototype solution will also be direct and not have extra unneeded features.

Section 1.2: Project Goals

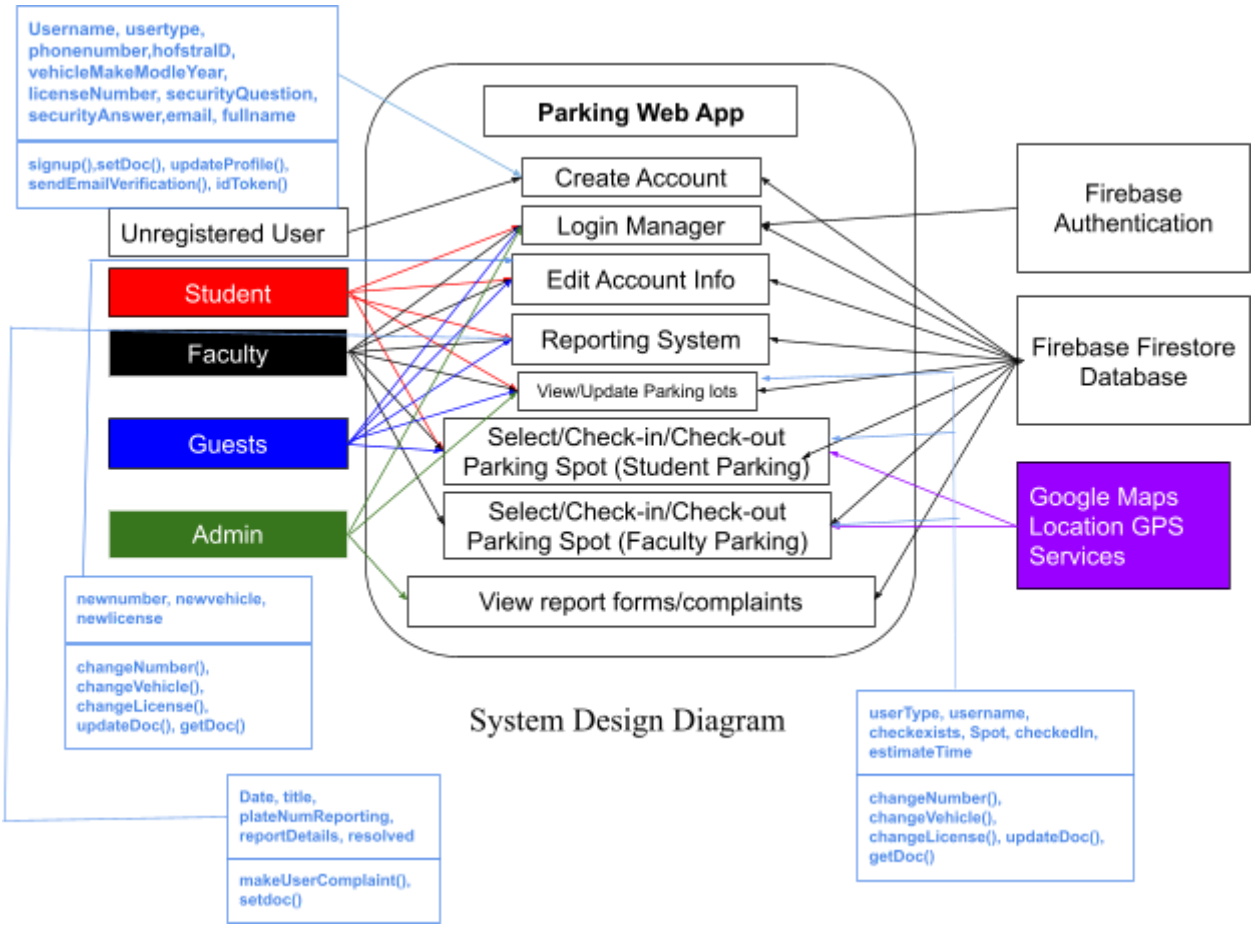
1. Create a web application that people can use to find parking easily and efficiently on Hofstra Campus.
2. The app will allow students and faculty to see if there is a parking spot available at the South or North Campus parking lot by showing maps with color coordination and numbering. These maps will be constantly updated.
3. This app will allow people to check-in by being in the vicinity of the spot. Location sharing/GPS will be used to counter reservations of spots.
4. The app will not allow students to check-in to a parking spot that is not available to their current user type along with parking closures.
5. The app will allow secure login to all users and will keep information on the user and their vehicle for admin personnel such as Public Safety in the case they need to contact the vehicle owner immediately or verify if the user is a registered hofstra.edu person (students, faculty, administrators). This will also let Public Safety know the guests that are visiting by car on campus.
6. The web app will provide a reporting system to resolve any issues users face while using the app.
7. The web app will require users to provide an estimated check-out time when checking-in in order to automatically check-out users that might forget to check-out of their parking spot.
8. The web app will be responsive for all phone types and sizes.

Chapter 2 Requirements Specification (RS)

This chapter details all requirements for our project. Section 2.1 describes the main features of our system and the interactions between our system, users, and other external systems. User stories describing the interaction between a user and the system are defined in Section 2.2. Section 2.3 describes the external interfaces and all inputs and outputs from the system. The functional requirements that describe the functions of the system and of its components are defined in Section 2.4. Section 2.5 describes the non-functional requirements and quality attributes of the system.

Section 2.1: System Perspective

Figure 1 is a diagram of the system showing interactions with users and external systems.



The system has a total of four users: student, faculty, guests, and admin.

Unregistered users can register as one of the five user types. Student and guest users can check-in a parking spot in only student designated parking spots. Student and guest users cannot check-in a faculty designated parking spot. Faculty users can check-in a student parking spot or faculty parking spot. Student, guest, and faculty users cannot see what an admin user sees. An

admin user can view a parking spot occupant's vehicle and user information and also view and resolve reports. User types that are handicapped will be able to check off whether they're handicapped or not when creating their account. Users that are handicapped will have the parking spots only available to handicapped users available to them along with the parking spots available to their user type. In the case that a user becomes handicapped, users can change their handicap status by contacting the admin either through the report system or in person.

Admin users can also update the availability of parking lots due to closures for university events.

There are two external components: database and Google Maps location services. The database will store, update, and send user information, grid maps, and reporting system. The Google Maps location services will aid in the check-in of parking spots and will tell if the user is on campus and in the parking lot of the available parking spot.

Section 2.2: User Stories

Name	US-1: Create User Account
Description	Users will input information based on the user type such as name, username, password, vehicle information, potential Hofstra identification number, and contact information to register for the app.
Rationale	Account registration is necessary for security and to utilize the web app functions.
Users	Unregistered user and all user types.
Preconditions	Faculty, student, and guest users must have a vehicle and identification number or pass.
Basic Course of Action	1. User clicks create account button. 2. System opens the registration form for general users. 3. User inputs user type (student, faculty, guest, and admin) 4. System opens the registration form for that user type. 5. User inputs all required fields in order to create an account including security questions. 6. System asks for identification and checks the user. 7. If successful, systems adds new user information including the user type and stores it in the database.
Alternative Paths	New methods to ensure the validity of user types.
Post Conditions	The system will have an updated database with the new user information.
Priority	5
Story Points	6
References	US-2

Name	US-2: User Login
Description	Users will login with their username and password set during registration to view parking lots and available parking spots to check-in and check-out.
Rationale	User must login to use the app's features and functions.
Users	All user types
Preconditions	User must have completed registration.
Basic Course of Action	1. User clicks login button. 2. System shows the username and password fields. 3. User inputs username and password. 4. System checks for valid username and password that is given. 5. If valid, system checks database for registered user. 6. If user is registered, the system allows the user to continue to the app.
Alternative Paths	If the user forgets their password then the user can click this button and the system will ask for the username and will ask the user the security questions used when the user registered for the account. If the user forgets the answers to the security questions, then they will need to contact the admin to change their password.
Post Conditions	User is able to access the app's features based on its user type.
Priority	4
Story Points	3
References	US-1, US-3

Name	US-3: Edit Account Information
Description	All registered users will be able to edit their account information once they are already logged into their account.
Rationale	In the case that the user wants to change password, username, contact information , and vehicle information.
Users	All user types except admin.
Preconditions	User must be logged in
Basic Course of Action	1. User clicks Edit Account 2. System shows Account Information retrieved from the database. 3. User makes changes and clicks submit. 4. System checks to see if input fields are valid and then updates the database.
Alternative Paths	User forgets their password before login.
Post Conditions	The system's database will be updated with the users new account information.
Priority	1

Story Points	3
References	US-1, US-2

Name	US-4: Report a Complaint
Description	All registered users will be able to report a complaint in the case they see an empty parking spot but it is checked-in on the app or if a user is parked in an undesignated spot.
Rationale	To allow users to report users who have not checked-out and are using a designated user's parking lot/spot. This feature allows users to report users misusing the app.
Users	All user types.
Preconditions	User must be logged in.
Basic Course of Action	1. User clicks Report button 2. System opens the form for reporting a complaint 3. User inputs required and optional fields and clicks submit. The user must make note of the license plate number. 4. System stores this complaint on the database to be retrieved by the admin.
Alternative Paths	N/A
Post Conditions	Admin users will be able to view the new complaints and act accordingly in order to solve the issue.
Priority	2
Story Points	3
References	US-2, US-8

Name	US-5: View Vacant/Occupied Parking Spots
Description	All registered users will use this feature to see the available parking spots left in certain parking lots by viewing grid maps with lot numbers and spot numbers that are color-coded by availability.
Rationale	To allow users to view what parking spots are vacant or occupied in a specific parking lot in order to help the user save time and find a parking spot fast.
Users	All user types
Preconditions	User must be logged in.
Basic Course of Action	1. User clicks View Parking spots button 2. System shows the different parking lots by name and picture. 3. User clicks on the parking lot they are looking for parking spots in. 4. The system shows the updated grid map of the parking lot with colors and

	numbers.
Alternative Paths	
Post Conditions	
Priority	5
Story Points	9
References	US-10

Name	US-6: Check-In/Check Out to Parking Spots According to user type
Description	Based on user type and location GPS sharing, users can check-in and check-out a parking spot.
Rationale	To ensure users park in their designated spots.
Users	A student user and guest user cannot check-in a faculty parking. A handicapped student/faculty user can check-into handicapped parking and parking available to their usertype. All users can only check-in if the location GPS shows they are on campus and near the vicinity of the spot.
Preconditions	User must have logged in.
Basic Course of Action	1. User views available parking spots. 2. User clicks on the parking spot that is available. 3. The system checks if a user can park at the spot based on user type and location GPS sharing checks if the user is on campus or in the parking lot. 4. The system shows pop-up action for the available parking spot. 5. User inputs estimated check-out time and then clicks check-in. 6. System updates database and the color changes to occupied.
Alternative Paths	
Post Conditions	The system has a new occupied parking spot and shows this change on the grid map and database.
Priority	5
Story Points	10
References	US-2, US-5, US-7

Name	US-7: Estimated Check-Out
Description	Users will have to provide an estimated time in which they will be expected to check out.
Rationale	This will prevent users from leaving their vehicles as checked-in for a spot on the app

	for multiple days in the case that they forget to check-out.
Users	All user types
Preconditions	Users must be logged in and already be in the process of checking-in for the parking spot they have parked their car.
Basic Course of Action	1. Once the user has parked their vehicles and checked-in to their parking spot 2. The system will ask the user for an estimated check-out time.
Alternative Paths	
Post Conditions	The system will update its database to include the estimated check-out time. If the user has been checked-in to the spot several hours after the estimated check-out time, the system will automatically check-out the user's vehicle from that parking spot.
Priority	2
Story Points	5
References	US-6

Name	US-8: View account information of all users parked in parking lot
Description	Admin users can see account information of all occupied parking spots and have access to the user database
Rationale	In the event a car needs to be moved or violates the app or to investigate reports, admin can make use of user information
Users	Admin users.
Preconditions	Admin must be logged in.
Basic Course of Action	1. Admin user clicks an occupied parking spot. 2. System shows the vehicle information and user information as well as the estimated 2check-out time of the checked-in car. 3. If the car is not checked-in, admin clicks on User Search by license plate number. 4. System retrieves information tied to the license plate.
Alternative Paths	
Post Conditions	
Priority	3
Story Points	6
References	US-2, US-5, US-6, US-7

Name	US-9: View report complaints
Description	Users of the user type admin will be able to view all the complaints other users have reported regarding the webapp and parking spots which will allow admins to resolve any problems that arise.
Rationale	This feature is needed in order to resolve any issues regarding the app.
Users	Admin users.
Preconditions	The admin user must be logged in.
Basic Course of Action	1. Admin user clicks on the view complaints button 2. The system retrieves the list of complaints inputted in the database and displays it for the user admin to view.
Alternative Paths	
Post Conditions	
Priority	1
Story Points	2
References	US-4

Name	US-10: GPS location sharing
Description	Users can only check-in at a parking spot if they are on campus or in the parking lot of the available spot.
Rationale	To avoid users reserving parking spots.
Users	All users except admin.
Preconditions	Users must allow GPS sharing on the web app in order to use the app's features.
Basic Course of Action	1. User attempts to check-in a parking spot. 2. System allows location GPS sharing to check location. If too far, system does not allow check-in. 3. System allows pop-up check-in if location is pass.
Alternative Paths	
Post Conditions	System allows a new check-in spot and updates the grid maps and database if location sharing passes.
Priority	5
Story Points	9
References	US-6

Name	US-11: Update parking lot for university closures and events
Description	User types of the status admin will be able to update parking lots in the case of university events or construction that needs a certain parking lot to be empty.
Rationale	With this feature, all users will see that they can't park in a certain parking lot in the case that the university needs that parking lot to be empty. This will prevent users from parking in spaces that the university needs to be empty.
Users	Admin
Preconditions	The admin user must be logged in.
Basic Course of Action	1. The admin user clicks on update parking lot view button. 2. The admin user inputs the parking to be closed. 3. The system will update the database to display the parking spaces in the parking lot to be not available.
Alternative Paths	
Post Conditions	The system will display an updated view of the parking spaces available in each parking lot.
Priority	2
Story Points	5
References	US-5

Section 2.3: External Interfaces

Section 2.3.1: User Interfaces

An external interface is Hofstra's parking lot maps to generate a user interface of grid maps. Another external interface is database and cloud service to manage and store grid maps information, user account information, and reporting information. The third external interface is Google Maps location services for user parking spot check-in.

Table 2.3.1: *Summary User Interfaces*

UI #	**UI Name**	**Description**	**References with user stories, reqs, constraints**	**Dependencies with other UIs**
UI-1	Registration	Webpage - Create Account button, form based on user type, submit button	US-1	
UI-2	Log-In	Webpage - Log-In button, username and password user input into input boxes, Forgot password and button	US-1, US-2	UI-1

UI-3	Forget password	Webpage - Forgot Password button under Login, Input for username, security questions, submit button	US-2, US-3	UI-2
UI-4	Edit Account Information	Webpage -Submit changes buttons, user input text box, labels showing the current account information	US-2, US-3	
UI-5	Report a Complaint	Webpage -Label(Title), user input textbox for title, label(description), user input text box for description, submit button, Label(Date), user input textbox for date, Label(license plate?), and user input text box.	US-4, US-5, US-8	
UI-6	View report Complaints	Webpage -List view of all recent user complaints.	US-9	UI-5
UI-7	View parking spots on grid maps	Webpage -Each parking lot in grid map view, color coded system to show vacant/occupied, and numbering system to represent parking spaces.	US-5	
UI-8	Check-In	Webpage -Pop-up with Label(estimated check-out time?), dropdown input for time), and button check-in	US-5, US-6	UI-7
UI-9	Check-Out	Webpage -Pop-up with label(Check-Out?), button(yes), and button(no).	US-6	UI-7
UI-10	View Account information	Webpage -Pop-up with labels of all the users account information.	US-8	
UI-11	Updated parking lot for university events/closures	Webpage -Each parking lot in grid map view, color coded system to show vacant/occupied, and numbering system to represent parking spaces.	US-11	

Section 2.3.2: Inputs and Outputs

Table 2.3.2.1: *Input Description*

Input #	**Name**	**Source**	**Description**	**Valid Range**	**Units of Measure**	**Data Format**	**References to user stories**
I1	Registration	text-field/button	Enter account information required fields, click submit button	8-20	Text length, Username criteria, Password criteria	text,button	U1, US-1
I2	Login	textfield/button	Enter username, enter password	8-20	Text length,	Char, int	US-2, UI-2

					Password criteria (1 uppercase, 1 special char)		
I3	Forget password	textfield/button	Enter new password into the text field. Enter answers to security questions in the text fields next to the question	8-20	Text length, Password criteria	Char, Int	US-2, UI-3
I4	Edit Account Information	textfield/button	Textfields that label each info: username, password, vehicle info, security questions, user type	8-50	Text length, license plate criteria, password citeria, username criteria	Char, Int, varchar	US-2, US-3
I5	Report a complaint	textfield/button	Text fields for users to input complaint title, description, date, and license plate number.	8-300	Text length, license plate number criteria	Char, Int, varchar	US-9, UI-5, UI-6
I6	Check-In	Button, drop down	Button to check-in, dropdown for the estimated check-out time.		Date/time format in scroll selection		US-5, US-6
I7	Check-Out	button	Label(check-out?), Yes button, no button				US-6
I8	Update parking lot for university events	Drop down menu, button	Dropdown menu consisting of all parking lots, submit button				US-11

Table 2.3.2.2: *Outputs Description*

Output#	Name	Destination	Description	Valid Range	Units of Measure	Data Formats	References to user stories
O1	Registratio	View	After the user registers, the view				U1, US-1

	n	Parking lot Webpage	parking lot interface shows up(Home page)				
O2	Log-In	View Parking lot Webpage	After the user logs in, the view parking lot interface shows up(Home page)				US-2, UI-2
O3	Forgot Password	View Parking lot Webpage	After the user enters all the fields to successfully reset the password, the view parking lot interface shows up(Home page)				US-2, UI-3
O4	Edit Account informatio n	Edit Account information	After the user clicks submit to submit changes a new version of the edit account information shows up with all the new information.				US-2, US-3
O5	Report a complaint	View Parking lot Webpage	After the user clicks submit to report a complaint the complaint gets added to the system and the user is redirected back to the home page.				US-9, UI-5, UI-6
O6	Check-In	View Parking lot Webpage	After the user clicks check-in, the user is directed to the same view parking lot webpage but will see their parking spot as occupied.				US-5, US-6
O7	Check-Out	View Parking lot Webpage	After the user clicks check-out, the user is directed to the same view parking lot webpage but will see their parking spot as vacant.				US-6
O8	Update parking lot for university events	View Parking lot Webpage	After the user chooses the parking lot to close and clicks submit, the user is directed to the view parking lot webpage but will see the parking spaces in the parking lot they closed as unavailable.				US-11

Section 2.4: Functional Requirements

Name	FR-1: Add user account
Description	System adds a new user after checking registration inputs.
Rationale	Users must be registered and linked to the database to access web app features
System Behavior	Registration needs to check user validity and input validity and store account information in the database.
References	US-1

Name	FR-2: Verify user account
Description	System checks login information
Rationale	Users must have valid login credentials
System Behavior	Check username and password combination in the database
References	US-2

Name	FR-3: Change User account information
Description	System updates account information if user submits changes
Rationale	User is able to change account settings and information and these changes need to be updated with the database
System Behavior	Easily update newly changed information; store and receive information to and from the database
References	US-3

Name	FR-4: View user account information
Description	System allows admin user types to view a parking spot occupant's account and contact information
Rationale	Admin is able to have special administrative duties and resolve reports and complaints.
System Behavior	The system should give access to admin users and keep current information on parking spots.
References	US-8

Name	FR-5: Add complaint
Description	System allows user to submit a complaint
Rationale	Users can report on issues and users misusing the app.
System Behavior	There should be no problems opening and submitting a report form.
References	US-4

Name	FR-6: View all complaints
Description	System allows admin type users to view all complaints submitted by users
Rationale	Admin have special administrative duties and resolve complaints
System Behavior	System should collect complaints in real time and be updated regularly
References	US-9

Name	FR-7: Check-In
Description	The systems will portray a spot to be occupied once a user checks-in to a parking spot.
Rationale	This requirement is needed in order for other users to see that a parking space is taken because a vehicle is now parked there.
System Behavior	The system's grid map must properly update its view to show the parking space is taken.
References	US-6

Name	FR-8: Check-Out
Description	The systems will portray a spot to be vacant once a user checks-out of a parking spot.
Rationale	This requirement is needed in order for other users to see that a parking space is available again because the vehicle once parked there left.
System Behavior	The system's grid map must properly update its view to show the parking space is available.
References	US-6

Name	FR-9: Update Parking lot for university events
Description	The system allows users of the type admin to update the view of the parking lots so that some parking spots are shown as unavailable.
Rationale	In the case that the university has construction or an event and needs a parking lot to be empty, the admin can use this to make sure no other users park in the area the university needs to be empty.
System Behavior	The Check-in and check out functions must work properly. The grid-map must update its view properly to show vacant and occupied spaces.

References	US-11

Name	FR-10: Admin User Search
Description	Users of the type admin will be able to search up other users based on their license plate numbers.
Rationale	When resolving reported complaints, the license plate of the person the other user made a complaint is given. The admin uses the license plate to look up the person with this function which will allow them to contact the person.
System Behavior	The database must be organized properly with all the correct information.
References	US-8

Name	FR-11: Estimated Check-Out
Description	System checks-out a vehicle automatically if the user hasn't checked out a certain time past the estimated check-out time the user gave when checking in.
Rationale	In the case that the user forgets to check-out, the parking spot won't be marked occupied for the whole day.
System Behavior	Check-out feature must work properly.
References	US-7

Section 2.5: Non-Functional Requirements

Section 2.5.1: Performance Requirements

NF-1: The web app's system will support a at least 20 simultaneous users
NF-2: The system will have a response time of 400ms.

Section 2.5.2: Logical Database Requirements

NF-3: The system's database will be able to store all the information required for at least 20 users including user account information, contact information, vehicle information, user status, security questions chosen and answers, and check-in and check-out times.
NF-4: The system's database will be able to retrieve data and update its contents based on user input.
NF-5: The system's database will be able to accurately send information to the web app to display updated grid maps and user information.

NF-6: The system's database will be able to easily access all the data for each user and make it accessible to admin users.

Section 2.5.3: Security Requirements

NF-7: Https used for all web requests
NF-8: Password protection for all users.
NF-9: Security questions required in account registration in the case of Forgot Password.
NF-10: Database only accessible to admin with authentication.
NF-11: Data integrity and confidentiality with user account information.

Section 2.5.4: Reliability Requirements

NF-12: 99% uptime of all end user facing features
NF-13: All user data backed up regularly

Chapter 3 Standards and Constraints

This chapter includes the standards and constraints that influence the development and implementation of this project.

Section 3.1: Standards

ST-1: Privacy Standards - This application must comply with the privacy standards set by ISO/IEC to ensure privacy in user's web and device usage. One specific standard is the ISO/IEC 19772:2020 (Authenticated Encryption). [22]

ST-2: Password Encryption and Hashing - This application must follow the password hashing and encryption standards provided by ISO/IEC 10118-1:2016 (Hash Functions). [24]

ST-3: Database Standards - The Parking Application must follow the SQL standards set by ISO/IEC in order to verify the storage of user data from the application to the database. A set of standards including foundation, framework, management of external data for the SQL Database are included in the ISO/IEC 9075 [23]

ST-4: HTTPS - HTTPS stands for Hypertext Transfer Protocol Secure and it's the primary protocol to send data between the web browser and website. In order to ensure security of data transfer, the HTTPS is encrypted, uses the secure socket layer, and certificate exchange system. Since the Parking App will utilize a login username and password along with requiring users to input personal information about themselves and their vehicle, the use of HTTPS is advised.

Section 3.2: Constraints

Section 3.2.1: Ethical

RC-1: The application and its features, as well the developers, will treat all persons fairly and with respect, to not engage in harassment or discrimination, and to avoid injuring others.
RC-2: Ensure confidentiality and privacy of data. User login profiles will be kept private from all persons and account information will be seen only by administrators.

Section 3.2.2: Legal

No major constraint or limitation.

Section 3.2.3: Economical

RC-3: Budget constraint is $200 for implementation.
RC-4: There are two developers on this project who are full-time students. This is a time cost constraint.

Section 3.2.4: Ergonomic and Aesthetic

RC-5: The user interface and web application must be responsive and easy to use on all forms of devices and for all users.

Section 3.2.5: Health

No major constraint or limitation.

Section 3.2.6: Safety

RC-6: The web application must ensure users are not driving while loading the web application through notifications and GPS location sharing.

Section 3.2.7: Welfare

No major constraint or limitation.

Section 3.2.8: Global

No major constraint or limitation.

Section 3.2.9: Cultural

No major constraint or limitation.

Section 3.2.10: Social

RC-7: The web application will help the Hofstra society in finding parking and lateness due to finding parking among members of society will decrease.

Section 3.2.11: Political

No major constraint or limitation.

Section 3.2.12: Manufacturing

No major constraint or limitation.

Section 3.2.13: Environmental

No major constraint or limitation.

Section 3.2.14: Sustainability

No major constraint or limitation.

Chapter 4 Project Design

This chapter includes the system design diagram and the detailed descriptions of each component and interactions between different components.

Section 4.1: Overview of System components

Figure 2 System Database Overview

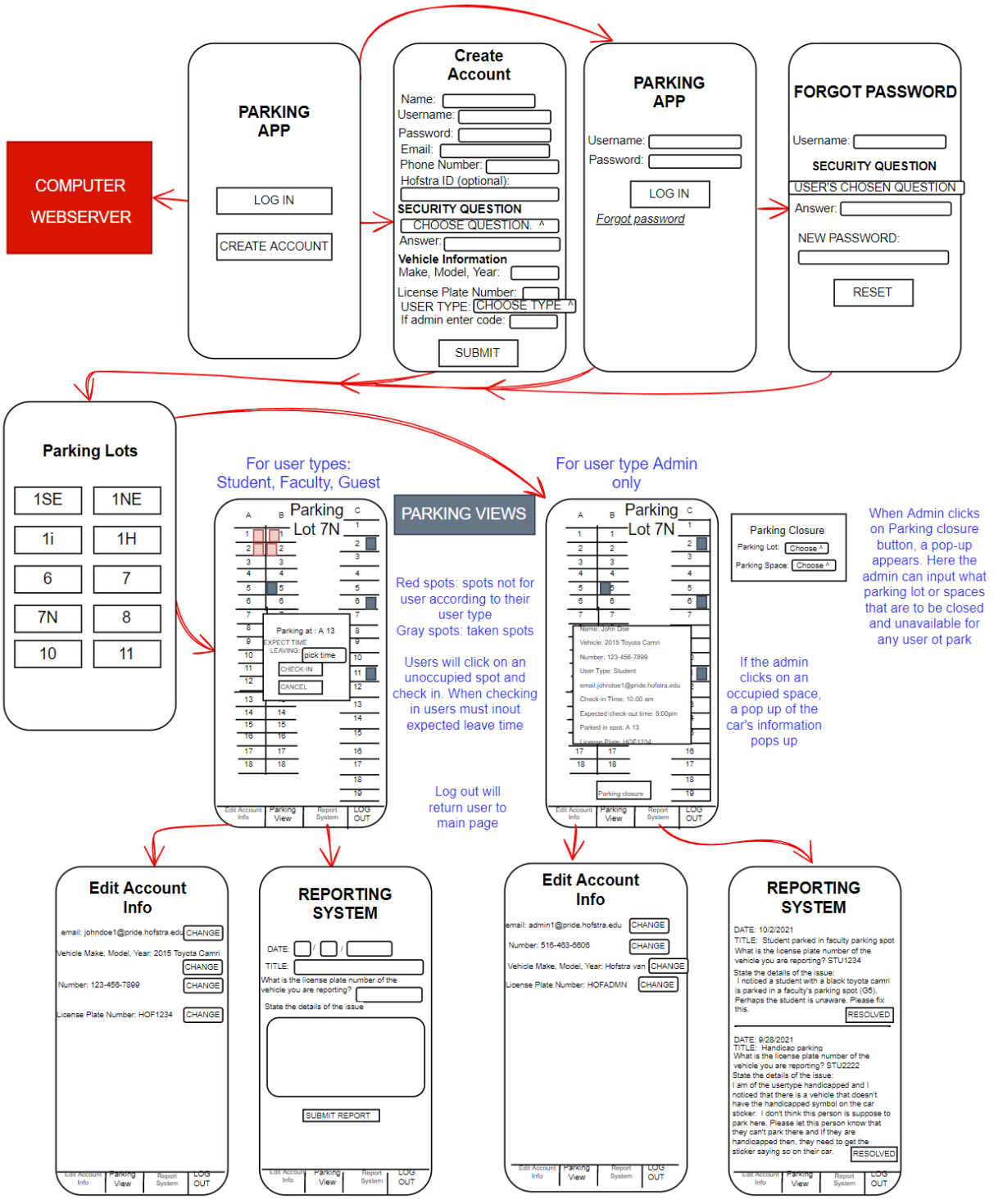


Figure 3 System Design Diagram

Parking App homepage is the first page displayed when the user opens the application with the options of logging in or creating an account. If the user clicks to create an account, the application will open up the create account page and ask the user to input all the information needed in order to create an account. On the other hand, if the user clicks to login, the application will open up to the login page asking the user to input the username and password. If users forget their password, they can click the forgot password link on the login page. This link will have the application open up the forgot password page where the user will have to answer the security question correctly in order to change the password.

The create account page, login page, and forgot password page will send the user to the Parking View of the application. Users will see the same view for the parking app view page but will have different actions available to them. The page for editing profile will be available for all user types to use in order to update any information displayed. The reporting system page will appear the same for users of type student, faculty, and guests where they will be able to report any problems regarding the application and parking spots. For example if a vehicle is parked in the wrong spot, another user can report that vehicle using the report system in the app.

In addition to that, there will be an estimated check-out time all users must input when checking into a parking spot. The application will automatically check out any vehicle that is parked 1 hour past their estimated check-out time. Users of type admin will only be able to view the complaint reports submitted by other users along with be able to click on the resolve button once the issue has been sorted out. All user types will be able to log out at any time while using this application by clicking on the log out button. After clicking the logout button the application will go back to displaying the homepage of the app where it asks users to either login or create an account.

Section 4.2: Structure and relationship

4.2.1 Parking App Homepage

The first component displayed is the Parking App homepage to all users. The options for two components are displayed on this page. Depending on whether the user already has an account, the user will choose between logging in and creating an account. Creating an account will be necessary for all unregistered users. Unregistered users will click to create an account and the application will open up the create account page. Registered users will click the login button to go to the login page.

4.2.2 Create Account

The create account page will require users to input their name, username, password, vehicle information, optional Hofstra identification number, user type, contact information, and will require the user to choose a security question and provide the answer to that question in order to register for the app (US-1). Unregistered users that are creating an account as an admin user will

need to input a code that will only be available by contacting the facilities using the app. The subcomponents of this page consist of the user's name, username, password, vehicle information, potential Hofstra identification number, user type, contact information, security question and its answer. The data will be passed into the database and will store the information as strings.

4.2.3 Log In

Registered users can log in by inputting their username and password into the login page of the application (US-2). The subcomponents in this component are the username and password. The application will use the database to verify whether the user has inputted the correct username and password.

4.2.4 Forgot Password

In the case that the user forgets their password or needs to reset their password, users can click on the forget password link. The forget password link will open up a page to reset the password however, users will be asked to choose the security question chosen when creating the account along with the correct answer to the question. The application will use the data stored in the database to verify whether the security question chosen and answer is correct. Users will only be able to change or reset their passwords if they choose the correct security question and input the correct answer. Once the user clicks the reset button, the password stored in the database will update to be the new password the user had input on this page.

4.2.5 Parking View Page

Once the user either created an account, successfully logged in, or changed their password, users will be sent to the main parking view page of the application. The main Parking View page will display all the parking spaces in the parking lot and will have a color coding system where the spots with a red box represent parking spots not available to the user based on their user type and the orange parking spots represent taken spots. Users of the type student, faculty, or guest will all have the same functionality for this page. The page will update a different view whenever other users check into a parking spot to show that the parking spot is now taken (US-5). The application will need the data stored in the database about the parking spots in order to display the spots taken or unavailable to the user. When the user has found a spot and parked in the parking spot, they will be able to click the parking spot they parked in on the app and check-into the spot. When checking into a spot, the user must also input the time they expect to leave the parking spot (US-7). General users will be involved with updating the check-in and expected check-out subcomponents when using this page (US-6). Once the user has checked into a parking spot, the data stored in the database regarding the parking spot will update. Users of the type admin have different actions accessible to them on the parking view page where they will be able to click on an occupied parking space and have all the vehicle's information appear (US-8). In order to do this, the database will output the values stored regarding the parking space the admin clicked on.

4.2.6 Edit Profile

In addition to that, all users are able to edit their account information on the edit account page on the application where they will be able to update/change their email, vehicle information, phone number, or license plate number (US-3). When a user changes a subcomponent, the data stored for that subcomponent in the database will also change.

4.2.6 Reporting System

Lastly, the parking application also has a reporting system accessible to all users and is used to report any issues or complaints users face while using the app or regarding parking spaces in general (US-4). The reporting system page will ask users to input the date, title of complaint, the license plate of the vehicle they are reporting, along with a short description of the complaint. If the user isn't reporting an issue about another vehicle and is reporting in regards to the usage of the app, providing a license plate isn't necessary. Once users have clicked the submit report button, the database will update to have the information regarding the complaint inputted. Just like the parking view page, admins have a different view of the reporting systems page where they simply see the information other users input when making the complaint (US-9). The subcomponents involved on this page are the date, title of complaint, the license plate of the vehicle they are reporting, along with a short description of the complaint. In order to have all the subcomponents involved on this page shown on the application, the database will output all the data stored in the database regarding all the complaints reported. After addressing the issue, admins can click on the resolved button to make the complaint disappear from the list of reported complaints displayed on the page.

Section 4.3: Detailed component description

Identification C-1	Create Account
Type	Module
Purpose	Account registration is needed for security and for the web app functions.
Inputs	name, username, password, vehicle information, Hofstra id, user type, contact information, and will require the user to choose a security question and provide the answer to that question in order to register for the app; Button click; **I1**
Output	User account stored to database; **O1**
Data	Data stored as strings
Dependencies	Login: username and password inputted from create account component is needed to verify user Forgot password: security question chosen and answer chosen from create account page used to verify user and allows user to change password Admin Parking View: the information the user inputted when creating an account appears when an admin clicks on the parking spot the user parked in.

Cross-Reference	I1, O1, U1, US-1

Identification C-2	Login
Type	Module
Purpose	To verify user and provide security for user's account information.
Inputs	**I2;** Enter username, enter password; Button Click
Outputs	**O2;** After the user logs in, the view parking lot interface shows up(Home page)
Data	Data stored as strings
Dependencies	Create Account: the user sets their username and password when creating an account.
Cross-Reference	I2; O2, US-2, UI-2

Identification C-3	Forgot Password
Type	Module
Purpose	Allow users access to their accounts and reset password in the case the user forgets password.
Inputs	**I3;** Enter a new password into the text field. Enter answers to security questions in the text fields next to the question
Outputs	**O3;** After the user enters all the fields to successfully reset the password, the view parking lot interface shows up(Home page)
Data	Data stored as strings
Dependencies	Login: When a user successfully changes their passwords, they need to use the new password to login. The new password replaces the old password in the database.
Cross-Reference	I3, O3, US-2, UI-3

Identification C-4	NonAdmin Parking View
Type	Module
Purpose	For users to check into parking spaces in order to notify other users that the parking space they parked in is now unavailable.
Inputs	**I2**
Outputs	**O2**
Data	Data is stored as strings

Dependencies	Create Account: The parking spots available to the user is based off of the usertype they registered as when creating the account. Admin Parking View: Admins will see the user's expected check out time along with seeing the exact spot the user checked-in.
Cross-Reference	I2, O2, US-2, UI-2

Identification C-5	Admin Parking View
Type	Module
Purpose	For administrative people to know information about vehicles parked at a specific parking spot. IN the case the spot needs to be empty, the admin can contact the vehicle's owner using this page.
Inputs	Click on parking spot
Outputs	Vehicle information
Data	Data is stored as strings and displayed on the page as labels.
Dependencies	Create Account: the vehicle information displayed relates to the information the vehicle owner inputted when making their account.
Cross-Reference	US-8

Identification C-6	Edit Profile
Type	Module
Purpose	Allow users to edit and update their account information
Inputs	**I4;** Text Fields that label each info: username, password, vehicle info, security questions, user type
Outputs	**O4;** After the user clicks submit to submit changes a new version of the edit account information shows up with all the new information.
Data	Data stored as strings
Dependencies	Users must be logged in. Admin's Parking View: if the user changes the email address and phone number on this page, the email and phone number displayed on the Admin's parking view will update too.
Cross-Reference	I4, O4, US-2, US-3

Identification C-7	Non Admin Reporting System
Type	Module

Purpose	To report any problems they're facing in regards to the application or the parking lots in general.
Inputs	**I5;** Text fields for users to input complaint title, description, date, and license plate number.
Outputs	**O5;** After the user clicks submit to report a complaint the complaint gets added to the system and the user is redirected back to the home page.
Data	Data is stored as strings and displayed on the page as labels.
Cross-Reference	I5, O5, US-9, UI-5, UI-6

Identification C-8	Admin Reporting System
Type	Module
Purpose	Allow Admin to view and resolve list of complaints
Inputs	Resolved button click on reporting system.
Outputs	NONE
Data	Data was stored as string and is displayed as labels.
Dependencies	Non admin reporting system: the complaint reports displayed on this page is based off of the reports inputted on the non admin reporting system page.
Cross-Reference	I5, US-9, UI-5, UI-6

Section 4.4 Methodologies

Section 4.4.1: Research new tools and methods

This section describes the research of new tools and knowledge needed to implement the project. Section 4.4.1 describes the tools such as programming frameworks, languages, and libraries researched. Section 4.4.2 describes the methods researched in terms of algorithms, processing methods, data representation and storage, and data communication.

Section 4.4.1.1 Research tools

Tool Name	RT-1: Svelte
Purpose	Javascript Compiler/Framework used to build dynamic and interactive web apps and user interfaces.
Advantages	Free. Fast, easy to use framework. It is written in typescript. Ability to write less code and use languages HTML, CSS, Javascript. "No Virtual DOM". Reactive.

Limitations	Unlike React, Svelte has a compile step. Fairly new framework which means less support, “lack of tooling”. Loose blueprinting. “Self-contained”.
References	[5][16]

Tool Name	RT-2: React
Purpose	Javascript Library/Framework for building user interfaces or UI components
Advantages	Free. Maintained by facebook and has support. Component-based, easily integrate other libraries and frameworks. Easy to learn. Option to use JSX syntax. Fast. Virtual DOM
Limitations	Focuses on the UI or the V in MVC architecture. JSX is a barrier.
References	[6]

Tool Name	RT-3: Angular JS
Purpose	Structural framework for dynamic web applications.
Advantages	Data binding. Create Rich Internet Applications (RIA). Provides developers to write client side apps using Javascript in a MVC architecture. Free and open-source. Dependency injection.
Limitations	Create Single Page Application. Not secure and need server side authentication and authorization. Not degradable if user disables javascript.
References	[7]

Tool Name	RT-4: MySQL
Purpose	Open Source database management system (RDBMS) and runs as a server. Uses the sql language.
Advantages	It is known for its speed, flexibility and reliability in web applications. MySQL database helps to automate data retrieving. Portability and performance. Known to be faster with read-only commands.
Limitations	MySQL does not support a very large database size as efficiently. Prone to data corruption. Lack of stability and scalability.
References	[8][10]

Tool Name	RT-5: PostGreSql
Purpose	Open source object-relational database system that is perfect for projects with object model domain.

Advantages	Known for reliability and robustness. PostgreSQL is known to be faster while handling massive data sets, complicated queries, and read-write operations. Support and documentation. Provides parameter security, as well as app security.
Limitations	Performance issues and backup recovery challenges.
References	[9][10]

Tool Name	RT-6: Google's Geolocation API
Purpose	Returns a location and accuracy radius based on information about cell towers and WiFi nodes that the mobile client can detect.
Advantages	Google's documentation is easier to understand; least learning curve.
Limitations	The learning curve. Not free but 5.00 USD per 1000.
References	[11]

Tool Name	RT-7: Google's Places API
Purpose	A service that returns information about places using HTTP requests. Places are defined within this API as establishments, geographic locations, or prominent points of interest.
Advantages	Useful in identifying parking lots: "The Places API uses a place ID to uniquely identify a place."
Limitations	Learning Curve and integration with the web app.
References	[12]

Tool Name	RT-8: Google's Geocoding API
Purpose	Process of converting addresses (like "1600 Amphitheatre Parkway, Mountain View, CA") into geographic coordinates (like latitude 37.423021 and longitude -122.083739), to place markers on a map or position the map.
Advantages	Use Geocoding API to find the address for a given place ID in conjunction with Places API. direct way to access these services via an HTTP request
Limitations	Learning Curve
References	[12][13]

Tool Name	RT-9: Amazon Location Service
Purpose	Add location functionality to applications without compromising data security and user privacy

Advantages	provides high quality maps, search for points of interest, addresses (geocoding), and lat/longs (reverse geocoding), routes, tracking, and geofencing.
Limitations	Free for 3 months. Learning Curve.
References	[14]

Tool Name	RT-10: Amazon Relational Database Service (RDS)
Purpose	Easy to set up, operate, and scale a relational database in the cloud
Advantages	Resizable capacity while automating time-consuming administration tasks such as hardware provisioning, database setup, patching and backups. Security and fast performance. Use PostGreSQL, MySQL, Oracle.
Limitations	Less customization and challenges with amazon support and unsupported features with MySQL
References	[15]

Tool Name	RT-11: Firebase
Purpose	Platform for creating web applications.
Advantages	Provides tools for tracking analytics, reporting and fixing app crashes, creating marketing and product experiment. Easy integration of location services by Google
Limitations	Learning curve and data migration
References	[17]

Section 4.4.1.2: Research methods

Method Name	RM-1: Password based authentication
Purpose	Passwords can be in the form of a string of letters, numbers, or special characters. Create strong passwords that include a combination of all options.
Advantages	The simple app only requires one factor authentication for speed and ease of use
Limitations	Less secure than multi-factor authentication
References	[18]

Method Name	RM-2: Firebase Authentication
Purpose	end-to-end identity solution, supporting email and password accounts, phone auth
Advantages	Fast implementation and comprehensive security

Limitations	
References	[19]

Method Name	RM-3: Cryptographic hash function (SHA-256)
Purpose	Algorithm that takes an input and returns hash value of 256-bits, or 64 hexadecimal digits to store into the database instead of the actual input.
Advantages	more secure than either MD5 or SHA-1
Limitations	“Performance-wise, a SHA-256 hash is about 20-30% slower to calculate than either MD5 or SHA-1 hashes”
References	[20]

Section 4.4.2: Identify new tools and methods

Section 4.4.2.1: Identify tools

RT-11 or Firebase is convenient and will have the least learning curve. Firebase also has RM-2 or authentication and includes Realtime Database which “provides an efficient, low-latency solution for apps that require synced states across clients in real time” [21]. Also RT-6 or Geolocation API from Google is the best choice as it is easily integratable and has a low learning curve.

Section 4.4.2.2: Identify methods

The methods to use are RM-1 and RM-2 based on the tools chosen in Section 4.4.2.1.

Section 4.5: Design alternatives

An alternate design to the parking view page of the parking application is to have the user simply input the spot they parked in along with display the number of the parking spots left in a given parking lot. The current design for this page is a grid system that visually displays the exact parking spots available and occupied. An advantage to switching to the newer design would be a neater and simpler way to store and show that a parking spot is taken. There wouldn’t be a need to check if the grid is displaying the correct information that the database has stored. However, a downside to this approach would be for the users because they wouldn’t be able to see exactly where they can go to park and would instead have to rely on the number of parking spots left in a parking lot.

In terms of code design, the MVC architecture fits best in this project with the different web pages/user interfaces, Javascript/HTML reacting to user input, and the model connecting with the server and database to provide input back to the user.

Section 4.6: Reuse and relationships with other products

No other projects' code and data will be reused.

Section 4.7: Resource list

Name	**Description**	**Availability**	**Cost**
Dr. Thomas Re	Faculty Advisor	2	None
Jumana Rahman	Developer/Student	8-10	None
Kavleen Kaur	Developer/Student	8-10	None

Section 4.8: Resource skill list

Table 4.8. List of Skills

Resource name	**Skill description**	**Where the skill will be applied**	**Reference to design components user stories**
Software engineering	Project design and testing, use of Svelte, MVC	Entire project	US-1...US-11
Mobile App	Use of MVC design	Entire Project	US-1...US-11
Ethics	Understanding the importance of user privacy and security on an application	Security and password + Account protection	C-2, C-3, US-2, UI-3
Databases	Use of MySql, relational databases, Query	Storing and receiving data, organizing data in database	C-1..C-8, US-1...US-11
Networking & Data Communications	HTTP requests, packets, performance	Rendering and hosting web app for multiple users	US-1...US-11
Web App	Use .NET core to develop a web application Use SQL server and stored procedures	Entire project	US-1...US-11
Intro to Cybersecurity	Password hashing and encryption algorithms	Security	C-2, C-3, US-2, UI-3
Ethical Hacking	Password security	Security	C-2, C-3, US-2, UI-3

Chapter 5 Test Plan

This chapter includes the test plans and test cases for the project.

Section 5.1: Test Plan

The test strategy for this project will be Unit Testing done at each end of the sprint cycles. There will also be integration testing done at the end of Sprints 2, 3, and 4 to test the working product as a whole. This will be done by manual testing. The main testing strategy for Sprint 1 will be unit testing. The connection between the website and the database will be tested by using the make account page and checking whether the new user has successfully been stored in the database. This unit test will be tested for each user type.

For Sprint 2, the functionality of other pages on the website such as the “make account” page, login page, “forgot password” page, and “edit profile” page will be tested by first conducting unit tests. The unit tests for the login page will consist of attempting an incorrect password multiple times to make sure a user with the incorrect password will be unable to login. Once we can ensure a user with an incorrect password can’t login, the correct password will be entered to test if a user with the correct login information can successfully login. The unit testing for the “forgot password” will be tested for each user type and will include resetting the password for a user using this page and then check on the database if the password has been changed. Unit testing for the edit “profile” page will be done by changing the contents about the user on the page and then checking the database to see if the contents about the user information has changed. This unit test will also test to see if the contents have successfully changed on the website too by logging out and back into the account. The integration and regression testing for the pages created in Sprint 3 will ensure that the “make account” page, login page, “forgot password” page, and “edit profile” page can all work together on the website. To test this a new account will be created and the login page will be used to enter into the account and then logout. Once this has been done, the “forgot password” page will be used to reset the password and the user should be able to logout and log into the account with the new password made. Lastly, the “edit profile” page will be tested by changing the user information contents on the page, logging out, and logging back in to see that the user information content has successfully been changed on the website.

For Sprint 3, unit tests will be performed for the different user types, parking check-in and check-out, the different categories of parking spots, and the accessibility of the admin user type to see all information. This will also test the database and its connection to ensure everything is being updated. Unit testing for the parking view page for general users will consist of successfully checking into a parking spot and checking out, along with ensuring an updated view of the parking view page every time someone checks in or out of a parking spot. Unit testing for the parking view page for admin users will consist of clicking on a parking space to check if the correct information about the vehicle parked in a spot is being displayed. In addition to that, the

parking closure feature for admins will be tested by closing a parking lot and checking whether the parking view has successfully updated on the website to show those parking spots as unavailable. Integration testing will be done so that the parking view for general user types and admin users will sync together with previous Sprint components.

For Sprint 4, unit testing will be performed for all cases of the reporting system for both normal users and admin. The unit tests for regular user types will include logging into an account, submitting reports on the reporting systems page, and checking if the reports have successfully been stored in the database. In addition to that, the unit testing for the admin reporting systems page will include logging into the admin account and clicking the resolve button to check whether the report solved has successfully disappeared from the reporting system page for the admin user type. Integration and regression testing for the reporting system feature will test to ensure that the reports made by regular users are visible to the admin user. Once the reporting system feature has been completed, more integration and regression tests will be conducted to ensure that the entire project works in all cases.

After all unit tests, integration tests, and regression tests have been completed, a complete system test will be conducted to ensure that the whole website is functional and works successfully.

Section 5.2: Test Cases

Table 5.2.1

Name	*TC - 1 Verify that make account works*
Type	Unit test
Description	“Make account” page must work properly for users to register.
Preconditions	Secure connection between database and website.
Basic course of events	1. User clicks “Make Account” in homepage 2. User is prompted to input text fields. 3. User input is stored in database.
Expected results	All user information regarding the new user is successfully stored in the database in an organized manner.
Acceptance criteria	Data of the new user is stored in the database and the user is successfully registered and sees the UI.
Cross-references	C-1..C-8, US-1...US-11
Scheduled Sprint	Sprint 1

Table 5.2.2

Name	*TC - 2. Verify that login works*
Type	Unit Test
Description	A registered user must be able to login and retrieve information from the database.
Preconditions	The user logging in has to have made their account already
Basic course of events	*1. Go to Login Page* *2.Enter incorrect username and try to login by clicking login button* *3. Enter incorrect password and try to login by clicking login button* *4. Enter the correct username and password and try to login by clicking login button*
Expected results	A user entering the incorrect username or password will be denied access to login to their account. A user with the correct username and password will be able login to their account
Acceptance criteria	User is able to view UI and has all information correctly displayed.
Cross-references	C-2
Scheduled Sprint	Sprint 2

Table 5.2.3

Name	*TC - 3 Verify that forgot password works*
Type	Unit Test
Description	User must be able to change password to retrieve account after successful authentication
Preconditions	The user trying to reset their password must have already made an account for the website.
Basic course of events	*1.Go to forgot password page* *2. Enter some correct information to the fields required leaving some fields blank* *3. Enter incorrect answers to some the fields required* *4. Enter all correct answers to all the fields required* *5. Change password to new password*

Expected results	The page will ask the user to complete all required fields when a required field is left empty and will display “incorrect answer” when the incorrect answer is given. Users will be denied access to change password until all correct answers to the security questions are inputted.
Acceptance criteria	A registered user’s new password is stored in the database for future login.
Cross-references	I2; O2, US-2, UI-2, C-2
Scheduled Sprint	Sprint 2

Table 5.2.4

Name	*TC - 4. Verify that edit profile works*
Type	Unit Test
Description	The user must be able to edit account information displayed in the “edit profile” page and changes must be shown in the database.
Preconditions	The user must have already made an account and successfully able to log into their account.
Basic course of events	*1. Login to account* *2. Go to edit profile page* *3. Change contents about user information*
Expected results	After changing the contents on the edit profile page, the website will display the new information the user had inputted when they changed the user information contents.
Acceptance criteria	Changes to the “Edit Profile” page are seen after logging back in.
Cross-references	C-6, I4, O4, US-2, US-3
Scheduled Sprint	Sprint 2

Table 5.2.5

Name	*TC - 5. Verify that parking view for general users works*
Type	Unit Test

Description	All general type users must be able to view available parking lots/spots and be able to check in and check out (if near) a parking spot.
Preconditions	User must have made an account and successfully logged in.
Basic course of events	*1. Click on parking spot that is not available* *2. Click on parking spot that is available* *3. Enter in some of the required fields to check into parking spot* *4. Enter all required fields to check into parking spot* *5. Check- out of parking spot*
Expected results	The website will display the reason why the user cannot check-into a parking spot if the user clicks on a parking spot that they aren't allowed to park in. The website will not allow a user to check into a parking spot until all required fields have been complete. The parking view will update the parking spot to be occupied once a user checks-into a parking spot. Depending on the time the user inputted for expected check-out time, the website will automatically check-out the user parked in the parking spot in the case someone forgot to check-out. When a user checks-out the parking view will update to show that the parking space is unoccupied.
Acceptance criteria	Successful check in and successful check out of a parking spot as well as accurate updated parking lot views. User can only check in if they are near a spot (GPS).
Cross-references	I2, O2, US-2, UI-2, US-8
Scheduled Sprint	Sprint 3

Table 5.2.6

Name	*TC - 6. Verify that parking view for admin works*
Type	Unit Test
Description	The admin user type must be able to see all parking spots checked-in and its information.
Preconditions	The user must have made an account and logged in
Basic course of events	*1. Click on a parking spot* *2. Click 'X' to exit out of vehicle information view for parking spot* *3. Click on Parking Closure button*

	4. Leave some fields blank. *4. Input all required fields for parking closure* *6. Click submit*
Expected results	When the admin clicks on a parking spot, all correct the user information about the vehicle parked in the spot should be displayed. The website will only allow the user to do a parking lot closure if all the required fields are displayed. Once the parking closure fields have been inputted and submitted the parking view should update to show the parking spots as “not allowed to park due to parking closure”
Acceptance criteria	Admin user type can view a parking spot’s contact number and name.
Cross-references	C-4, C-5, I2, O2, US-2, UI-2, US-8
Scheduled Sprint	Sprint 3

Table 5.2.7

Name	*TC - 7. Verify that reporting system for general users works*
Type	Unit Test
Description	A general user must be able to successfully submit a report about a parking spot/car.
Preconditions	User must have made account and logged in
Basic course of events	*1. Go to Reporting Page* *2. Try to submit a report leaving some required fields blank* *3. Submit a report completing all required fields* *4. Submit report on the reporting systems page*
Expected results	The website will only submit a report once all fields are correctly inputted. New report will be stored in the database.
Acceptance criteria	General users see the submitted button and the admin user type sees the report.
Cross-references	C-8, UI-6, I5, US-9, UI-5, UI-6
Scheduled Sprint	Sprint 4

Table 5.2.8

Name	*TC - 8. Verify that reporting system for admin works*
Type	Unit Test
Description	The admin user type must be able to view new reports and resolve these reports.
Preconditions	Admin is successfully logged in.
Basic course of events	*1. Database fetches most recent and updates list of unresolved reports* *2. Resolve button makes the report disappear*
Expected results	After Admin logs in, the “reporting system” page must have an updated list of unresolved reports. The resolve button next to each report makes the report disappear.
Acceptance criteria	Admin sees updated, most recent reports and resolve issue works.
Cross-references	C-8, UI-6, I5, US-9, UI-5, UI-6
Scheduled Sprint	Sprint 4

Table 5.2.9

Name	*TC - 9. Verify integration between “make account”, login page, “forgot password”, and “edit profile” page*
Type	Integration and Regression Test
Description	The user must be able to edit account information using “make account”, login page, “forgot password”, and “edit profile” page. The user must be able to login using updated information and view updated information on the “edit profile” page.
Preconditions	The user must have already made an account and successfully able to log into their account.
Basic course of events	*1. Login to account* *2. Logout* *3. Complete all required fields to forget password and reset password* *4. Logout* *5. Login using new password* *6. Go to edit profile page*

	7. Change user information on the edit profile page *8. Logout*
Expected results	The user will be able to use information inputted when making an account to log into their account. The user will be able to use the new information inputted in the system when completing forgot password and edit profile page.
Acceptance criteria	All information changed must be updated in the database.
Cross-references	C-1..C-8, US-1...US-11
Scheduled Sprint	Sprint 2

Table 5.2.10

Name	*TC - 10. Verify integration between "parking view" page for general users and admin*
Type	Integration and Regression Test
Description	Any change the admin makes on the parking view page must be displayed on the parking view page for general users.
Preconditions	The user must have already made an account and successfully able to log into their account.
Basic course of events	*1. Login to admin account* *2. Make a parking closure* *3. Logout* *4. Log into general user* *5. Try to check into a parking spot the admin had closed*
Expected results	The parking view must update for all user type when an admin makes a parking lot closure. The parking view must update for all users when a user checks in or out of a parking spot.
Acceptance criteria	All information changed must be updated in the database.
Cross-references	C-8, UI-6, I5, US-9, UI-5, UI-6
Scheduled Sprint	Sprint 3

Chapter 6 Project Plan

This chapter includes the Project Plan and four AGILE Sprints for the implementation of this project.

Section 6.1: Sprints
Section 6.1.1: Sprint 1

- **Goal**: Establish a secure connection with the database the web application is using. The website should be able to successfully send and receive data to the database. The database should successfully store all the data without losing any data.
- **References**: C-1, C-2, I1, O1, U1, US-1, I2, O2, US-2, UI-2, RC-5
- **Starting date**: January 3, 2022 **Duration**: 4 weeks
- **Sprint master**: Kavleen Kaur
- **Demo goal**: The demo will show the create account page of the website. We will demonstrate that there is a secure connection between the database and the website by creating a new user and displaying how the new user is stored in the database. We will demonstrate the user interface of all the webpages in the application except the parking grid page.

Table 6.1.1: *Backlog list for a Sprint 1*

Task #	Task description	Hardware/Software resources	People(s)	Estimated hours	Points difficulty	Cross-reference
1	Create a project and all webpages (interface layout) except parking grid	Visual Studio Code,	Jumana and Kavleen	20	5	US-1...US-11
2	Research how to create & link Database to website		Jumana and Kavleen	12	8	C-1..C-2, US-1...US-2
2	Create Database and organize data entities in database	Firebase, Microsoft SQLServer	Kavleen	5	7	C-1..C-8, USf-1...US-11
3	Link and make a connection between project and database	Visual Studio Code, Microsoft SQLserver, Firebase	Kavleen	5	8	C-1..C-8, US-1...US-11

4	Test the connection by making new user		Kavleen	5	8	C-1..C-8, US-1...US-11

Section 6.1.2: Sprint 2

- **Goal**: Creation of user account, Account verification in login page, account verification and successful password change in forgot password page, allow users to successfully edit contents on the edit profile page, and logout.
- **References**: C-1, C-2, I1, O1, U1, US-1, I2, O2, US-2, UI-2, RC-5, C-3, I3, O3, UI-3, C-6
- **Starting date**: Feb 1, 2022 **Duration**: 4 weeks
- **Sprint master**: Jumana Rahman
- **Demo goal**: The demo will demonstrate a user creating an account, logging in, and changing contents of user information on the edit profiles page. Then we will demonstrate the case in which the user forgets password by logging out and going to the forgot passwords page. After doing so, we will demonstrate that the forgot passwords page was successful by logging into the account with the new password.

Table 6.1.2: *Backlog list for a Sprint 2*

Task #	***Task description***	***Hardware/Software resources***	***People(s)***	***Estimated hours***	***Points difficulty***	***Cross-reference***
1	Ensure the all the components in create account page works and is stored in the database	Firebase, Microsoft SQLServer, Visual Studio Code	Kavleen	5		C-1, C-2, C-3
2	Apply encryption methods to secure user information in the database		Kavleen	10		I2; O2, US-2, UI-2, C-2
3	Use the database to check information to allow secure login from user.	Firebase, Microsoft SQLServer, Visual Studio Code	Jumana	5	5	C-2

4	Successful logout	Firebase, Microsoft SQLServer, Visual Studio Code	Jumana	5	4	C-2
5	Functionality of edit profile page. Make sure the contents changed on this page also changed in the database.	Firebase, Microsoft SQLServer, Visual Studio Code	Jumana	10	6	C-6, I4, O4, US-2, US-3
6	Functionality of forgot password page. Verify users with security questions. Login with the new password to check if it is successful.	Firebase, Microsoft SQLServer, Visual Studio Code	Kavleen	10	8	I2; O2, US-2, UI-2, C-2

Section 6.1.3: Sprint 3

- **Goal**: Complete the user view and admin view of the parking lot page, complete the functionality for all user types(check-in, check out, estimated check-out, unavailable parking spots), functionality of admin parking closure, and functionality of admin parking spot detailed information pop-up.
- **References**: C-4, C-5, I2, O2, US-2, UI-2, US-8
- **Starting date**: March 1, 2022 **Duration**: 4 weeks
- **Sprint master**: Kavleen Kaur
- **Demo goal**: The demo will demonstrate the main functionality of the parking app. We will already have an account created for the three different user types and login with each different user type. We will demonstrate the different views for the different user types by showing that the parking spots available for each user are different based on their user type. We will then demonstrate how a user will check-into a parking spot using the parking lot we are nearest to and after checking in, the webpage should update to show that spot is now taken. This will be done to demonstrate the check-out feature too. Once we demonstrate each user type checking into a parking spot, we will log into the admin's account and display what the admin sees along with demonstrating all the functionalities only accessible to admin(view all the details of the parking spot and parking lot closures).

Table 6.1.3: *Backlog list for a Sprint 3*

Task #	***Task description***	***Hardware/Software resources***	***People(s)***	***Estimated hours***	***Points difficulty***	***Cross-reference***
1	Research and create user interface for parking lot views	Firebase, Microsoft SQLServer, Visual Studio Code	Kavleen and Jumana	10	10	C-4, C-5
2	Different parking spots available depending on usertype	Firebase, Microsoft SQLServer, Visual Studio Code	Kavleen and Jumana	25	10	C-4, C-5
3	Working Check-in/Check-out for all user types.	Firebase, Microsoft SQLServer, Visual Studio Code	Jumana	8	6	I2, O2, US-2, UI-2, US-8
4	GPS	Firebase, Microsoft SQLServer, Visual Studio Code	Jumana	5	5	I2, O2, US-2, UI-2, US-8
5	Automatic check-out for users using estimated check-out time	Firebase, Microsoft SQLServer, Visual Studio Code	Jumana	3	4	US-2, UI-2, US-8
6	Admin view of parking lot	Firebase, Microsoft SQLServer, Visual Studio Code	Kavleen	10	9	C-5
7	Fetch data from database and display when admin clicks on a parking spot to display all info on vehicle and car owner.	Firebase, Microsoft SQLServer, Visual StudFirebase, Microsoft SQLServer, Visual Studio Codeio Code	Jumana	8	5	C-4, C-5, I2, O2, US-2, UI-2, US-8
8	Functionality of admin parking closures pop up.	Firebase, Microsoft SQLServer, Visual Studio Code	Kavleen	15	10	US-2, UI-2, US-8
9	Verify if admin parking lot closure works by logging into a user account and seeing if the parking lot closed is available to park in	Firebase, Microsoft SQLServer, Visual Studio Code	Kavleen	10	9	C-4, C-5, I2, O2, US-2, US-8

Section 6.1.4: Sprint 4

- **Goal**: Develop a working reporting system accessed and usable by all users of the application including admin and normal users. Reporting system needs to be shown on the user interface by administrators who can then resolve the reports. Finish any other features and touch-ups.
- **References**: C-7, I5, O5, US-9, UI-5, C-8, UI-6, I5, US-9, UI-5, UI-6
- **Starting date**: April 1, 2022 **Duration**: 4 weeks
- **Sprint master**: Jumana Rahman
- **Demo goal**: The demo will show a user who will generate a report on a car/parking lot. The demo will then show an admin login, who will see the reporting system page with the new report. The admin can resolve this report and once the report is resolved, depending on if a specific license plate was given, a count for the amount of times the vehicle has been reported will increase by one. This count will be hidden from the user and just stored in the database for other use.

Table 6.1.4: *Backlog list for a Sprint 4*

Task #	*Task description*	*Hardware/Software resources*	*People(s)*	*Estimated hours*	*Points difficulty*	*Cross-reference user stories, require-elements, design*
1	Ensure all UI components of the Reporting System page works	Firebase, Microsoft SQLServer, Visual Studio Code	Jumana and Kavleen	8	5	C-7, I5, O5, US-9, UI-5
2	Create New Report	Firebase, Microsoft SQLServer, Visual Studio Code	Jumana	5	4	C-7, I5, O5, US-9, UI-5,
3	Reports link to database; send + fetch	Firebase, Microsoft SQLServer, Visual Studio Code	Jumana	5	5	C-8, UI-6, I5, US-9, UI-5, UI-6
4	Admin sees new reports	Firebase, Microsoft SQLServer, Visual Studio Code	Kavleen	5	5	C-8, UI-6, I5, US-9, UI-5, UI-6
5	Resolve reports & increase count (if a vehicle was reported)	Firebase, Microsoft SQLServer, Visual Studio Code	Kavleen	5	5	C-8, UI-6, I5, US-9, UI-5, UI-6

Section 6.2: Risk Plan

RP-1: Task List/Time

a. **Risk Description:** The task list is detailed however there will be a learning curve to bring this all together. This risk identifies the time of research/learning versus the time of implementation and that there may not be time to fully implement all components.
b. **Likelihood:** 5/10
c. **Impact:** 4
d. **Actions to alleviate:**
    i. Use January to get a head start to research and plan.
    ii. Implement and stick to a strict time schedule per week and month.

RP-2: Database Connection

e. **Risk Description:** The database and web application take too long to communicate and render information about parking lot spots.
f. **Likelihood:** 7/10
g. **Impact:** 5
h. **Actions to alleviate:**
    i. Use January to fully configure the database and web application.
    ii. Research and plan before creating a database and web application.

Section 6.3: Estimated Financial Budget

Table 6.3: *Budget estimate*

Item	Estimated Cost
API	$50.00
Web server rentals or AWS	$50.00
Google GPS and Maps	$50.00

Section 6.4: Teamwork Plan

For the completion of this project, the developers will use Jira, a project management software, to record task assignments and other information for the progress of the application. The developers will have two 30 minute meetings on Monday and Friday every week. This will ensure accountability for the week and task assignments get done. The developers will use Github for version control and code source.

List of References

[1] Park Mobile for College Campuses available at ***https://parkmobile.io***

[2] Park Mobile's Frequently Asked Questions available at ***https://support.parkmobile.io***

[3] East Tennessee Sata App Helps Students Hunt Down Parking Spots available at ***https://campustechnology.com***

[4] Best Parking available at ***https://www.bestparking.com***

[5] Svelte framework available at ***https://svelte.dev/***

[6] React framework available at ***https://reactjs.org/***

[7] AngularJS framework available at ***https://angularjs.org/***

[8] MySQL available at ***https://diliru.medium.com/advantages-and-disadvantages-of-using-mysql-36f6ffce3fa3***

[9] PostGreSQL available at ***https://www.aalpha.net/blog/pros-and-cons-of-using-postgresql-for-application-development/***

[10] PostGreSQL vs MySQL available at ***https://www.xplenty.com/blog/postgresql-vs-mysql-which-one-is-better-for-your-use-case/***

[11] Geolocation API available at ***https://developers.google.com/maps/documentation/geolocation/overview***

[12] Places API available at ***https://developers.google.com/maps/documentation/places/web-service/overview***

[13] Geocoding API available at ***https://developers.google.com/maps/documentation/geocoding/overview***

[14] Amazon Location Service available at ***https://aws.amazon.com/location/***

[15] AWS RDS available at ***https://aws.amazon.com/rds/?nc2=h_ql_prod_fs_rds***

[16] Web Apps with Svelte available at ***https://dev.to/kalashin1/building-web-apps-with-svelte-4bj7***

[17] Firebase Platform available at ***https://firebase.google.com/***

[18] Authentication Methods available at ***https://www.idrnd.ai/5-authentication-methods-that-can-prevent-the-next-breach/***

[19] Firebase Authentication
https://firebase.google.com/products/auth?gclid=CjwKCAjw_L6LBhBbEiwA4c46upMKIuvHeR9JcJfk-hUZuxqGt65aNWcScqJtailf-xw4cGRUa9l9xRoCaVIQAvD_BwE&gclsrc=aw.ds

[20] Hash Algorithm available at
https://www.freecodecamp.org/news/md5-vs-sha-1-vs-sha-2-which-is-the-most-secure-encryption-hash-and-how-to-check-them/

[21] Realtime Database Firebase available at
https://firebase.google.com/docs/database/rtdb-vs-firestore?authuser=0

[22] Webstore Privacy Standard available at
https://webstore.ansi.org/industry/software/Encryption-Cryptography

[23] Webstore Software Engineering Standard

https://webstore.ansi.org/industry/software/software-engineering#SQL

[24] Webstore Password Hashing Standards available at
https://webstore.ansi.org/industry/software/Encryption-Cryptography/Hash-Functions

[25] IEEE Code of Ethics

https://www.ieee.org/about/corporate/governance/p7-8.html

[26] Code of Ethics

https://www.acm.org/code-of-ethics