Zombie Apocalypse Prep App

Jorge Guzman-Nader - (guzmannj)
Christa Wright - (wrighch3)
Blake Hudson - (hudsonbl)
Eric Sisson - (sissone)
Kuan-Yu Lai - (laik)

January 27, 2018

1 Project Description

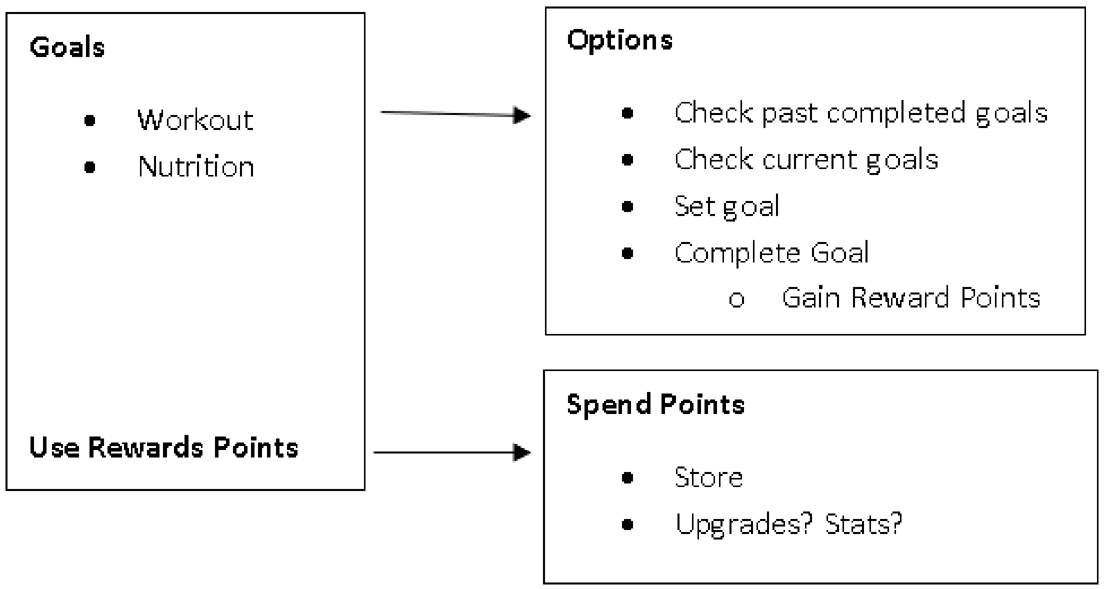
1.1 What is our approach, and who are our target users? How does our system help them?

Our target users are those who want a healthy lifestyle, fans of the zombie genre, and those looking to be prepared for a zombie apocalypse. Our application will help users obtain a healthier lifestyle by setting goals for the user and motivating the user through a reward system. The application will also help users get their post-apocalyptic fix through the zombie based scenario.

1.2 State the key difference between your approach and previous approaches?

A similar android app called "zombie run" addresses the use of a simulated zombie apocalypse to make users exercise. Our app differs to the previous one in that it can be used indoor, that is without the need to move from one location to another, because it does not employ geolocation.Another functionalities of our app include: a nutrition planner feature, a workout buddy searching capability.

1.3 Diagram to show how our approach plugs into exiting resources?



The diagram above shows an outline of the program we are making. The main features that we want to approach first are shown here. These features include setting goals, completing goals, checking current and past goals, and using reward points from the completed goals to get in-game rewards from the store.

1.4 Useful Programming Languages and tools used for this project

The programming languages and tools that will be used in the project are HTML, CSS, JavaScript, Ruby on Rails, Bootstrap and GitHub. We discerned that making a web application will be the easiest way to approach this program, so HTML, CSS, and JavaScript are the most appropriate programming languages for this project. Ruby on Rails and Bootstrap a free open source software that is designed for modern web applications, making them appropriate tools for this project. Github will be used to help organize everything and allow members to access the files within the project.

1.5 What are the functional requirements?

Our web application needs to be mobile friendly, it will set a goal list, that is modifiable by the user, and that give points or rewards when the user accomplish one action in the list, such as running one mile.

It will also require input of weight, height, gender, age and activity level for the nutrition feature of the app, then the program will provide output accordingly.

1.6 What are its non-functional requirements?

The web app should be able to accept upgrades, accomplished goals should be stored into the history option of the app.

1.7 What other documentations will we provide to help our users understand and use our system?

There will be no outside documentations to help users understand our system. There will be a help button that will describe what actions can be done in the application, and what they do. We are potentially thinking of implementing a tutorial, but we believe a help button will suffice.

1.8 What are the major features of the system that make it so effective and useful?

The major features of the application include the ability to set workout and nutrition goals, and the ability to track the progress of the user. These features will help users obtain a healthier lifestyle and is effective due to the reward system that will motivate users. We are tentative on including a Gym Pal system which will help users find partners for their workouts. While this is another motivation for users to stay healthy, we want to focus more on personal goals and workouts for the user.

2 Use Cases

A user can be looking to become more healthy by setting healthy goals. How does a person accomplish these goals. By setting goals on our online application, users are rewarded for accomplishing workouts. Consistency is key to accomplish any difficult task. Working out is a difficult task. So why not make working out fun?

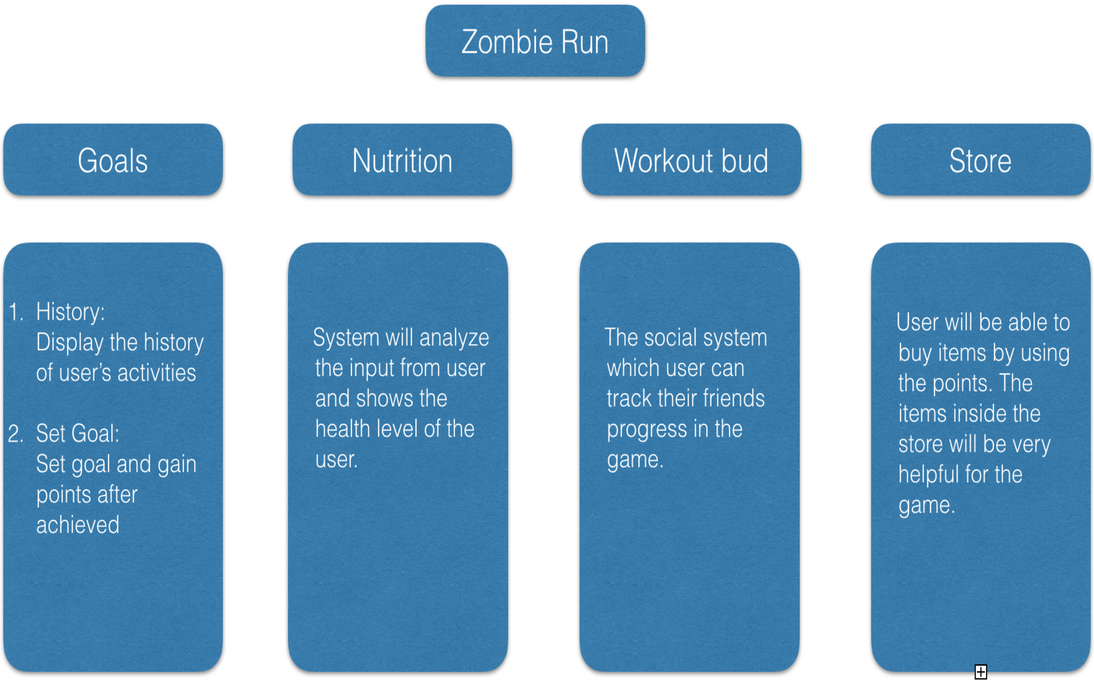
Flow of events:

- Main menu displays Goals, Nutrition, Workout Bud, and Store
- To set a goal, user selects goals

- Win points for accomplishing goal
- Spend points from accomplishing goals
- Acquire more in-store items by setting more goals
- View history of accomplishments
- Personally benefit from making healthy lifestyle choices!

Quality Requirements:

- Access to our online web application via computer, phone, tablet
- To accomplish goals the user will enter in data from their accomplishments(runs, workouts, etc.).



Zombie Run

Goals

1. History:
   Display the history of user's activities
2. Set Goal:
   Set goal and gain points after achieved

Nutrition

System will analyze the input from user and shows the health level of the user.

Workout bud

The social system which user can track their friends progress in the game.

Store

User will be able to buy items by using the points. The items inside the store will be very helpful for the game.

2.1 Scenario 1:

Name: Setting a workout and nutritional goal

Goal Level: Allow the user to set physical or nutritional goals for themselves that will produce rewards when completed.

Actor: Application user.

Preconditions:

- User uses/clicks the "set goal" option
- A goal is now created and the user can work to finish it

Postconditions:

- Once the user finishes the goal, they enter the information from their goal. This can be the number of miles ran or number of push ups

- The goal will be recorded and can be checked by the user.
- the user will be rewarded for achieving the goal by points
- A reward will be generated and listed on the goal set by the user. This reward will be given when the user completes the goal.

2.2 Scenario 2:

Name: Check History

Goal: Allows the user to check a list of past goals that have been completed by the user.

Actors: Application user

Precondition:

- The user uses/clicks the "Goals" option, then "history"

Postcondition:

- A list of past activities
- The user now can create new goals based off of past activities

2.3 Scenario 3:

Name: Store

Goal: Users will be able to purchase items by using the points they have earned. The items inside the store will increase the survival rate of the user.

Preconditions:

- User selects store
- User purchases a item from the in game store
- User attempts to purchase another item they do not have enough points for

Postcondition:

- User can use the items they bought
- User is alerted that they do not have enough coin and must set more goals for more rewards

2.4 Why these scenarios are helpful

The three scenarios are the most important because they discuss how the user used different core parts of our application. Goal setting is our primary objective for users to begin doing for themselves in their every day lives.

Scenario 1 describes the actor setting a workout and nutritional goal. The workout and nutritional goals themselves are separate options within the main user interface. Once the actor selects either goal it is up to them to complete their goal. Whether it is to go on a 2 mile run, or 100 push ups, or eating nutritionally healthy. When the user finishes their goal, they enter in the data from the workout and then they are rewarded points!

Scenario 2 describes how the user can view their recent goals. By viewing recent goals, this will help users create new goals to be achieved.

Scenario 3 describes the actor utilizing our in game store! In this scenario, the user buys an item from our in game store. The purchase was successful and the reward was awesome! But, the actor or user, ran into a situation where they don't have enough points to purchase an item. They must accomplish more goals in order to acquire more points and in return, purchase more rewards!

2.5 Error Scenarios

Since the premise of our game is based on users setting their own goals. Then the user enters their own data such as how many miles they ran. So the honor system will be upheld to our users. Potentially our users can just sit on the couch and enter in that they went on a 50 mile run. A few ways around this is by setting limits on the data users enter and how often they are able to enter in new goals.

3 Planning

3.1 Milestones & Tasks

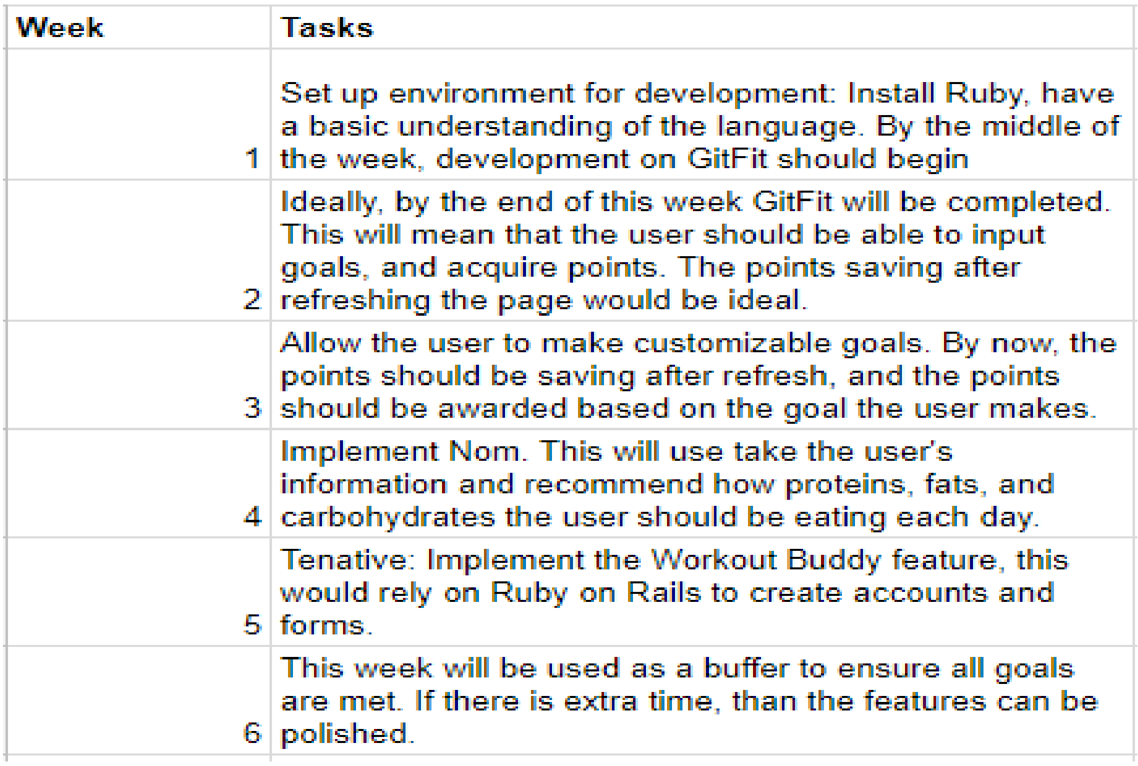
To begin building the Zombie Apocalypse Prep App, we will split the project into the three key features; GitFit, Nom and GymPal. Before beginning development, we will ensure each individual has the correct environment setup for development.

The first feature is the most important to the development of the app would be GitFit. This would require an option for a goal to be generated, and when the goal is completed, or checked off an arbitrary amount of points are rewarded, and added to whatever number is currently awarded. Ideally the goal will be customizable. The interface will need to be 'lightly' developed to allow a goal to be made, and removed upon completion.

The second feature, seems straightforward to program. It would rely on the user entering numeric information about themselves, and the program would generate nutritional recommendations to them, such as height and weight. It would be based on an equation that takes the user's h The program would suggest how much protein, fat, and carbohydrates the user ought to eat each day. A forum would need to be written to allow the user to input the information, and background logic would be used. Lastly, a table would need to written with the correct information displayed.

The last feature, GymPal, would require accounts to be made for users to communicate in a forum. The system would rely on Ruby on Rails to create accounts, and allow users to create posts and comments.

3.2 Schedule and Timeline



Week	Tasks
1	Set up environment for development: Install Ruby, have a basic understanding of the language. By the middle of the week, development on GitFit should begin
2	Ideally, by the end of this week GitFit will be completed. This will mean that the user should be able to input goals, and acquire points. The points saving after refreshing the page would be ideal.
3	Allow the user to make customizable goals. By now, the points should be saving after refresh, and the points should be awarded based on the goal the user makes.
4	Implement Nom. This will use take the user's information and recommend how proteins, fats, and carbohydrates the user should be eating each day.
5	Tenative: Implement the Workout Buddy feature, this would rely on Ruby on Rails to create accounts and forms.
6	This week will be used as a buffer to ensure all goals are met. If there is extra time, than the features can be polished.

To begin building the Zombie Apocalypse Prep App, we will split the project into the three key features; GitFit, Nom and GymPal. Before beginning development, we will ensure each individual has the correct environment setup for development.

The first feature is the most important to the development of the app would be GitFit. This would require an option for a goal to be generated, and when the goal is completed, or checked off an arbitrary amount of points are rewarded, and added to whatever number is currently awarded. Ideally the goal will be customizable. The interface will need to be 'lightly' developed to allow a goal to be made, and removed upon completion.

The second feature, Nom seems straightforward to program. It would rely on the user entering numeric information about themselves, and the program would generate nutritional recommendations to them, such as height and weight. It would be based on an equation that takes the user's h The program would suggest how much protein, fat, and carbohydrates the user ought to eat each day. A forum would need to be written to allow the user to input the information, and background logic would be used. Lastly, a table would need to written with the correct information displayed.

The last feature, GymPal, would require accounts to be made for users to communicate in a forum. The system would rely on Ruby on Rails to create accounts, and allow users to create posts and comments.

3.3 Project tracking

We will use GitHub to monitor the progress of the app. By using GitHub we can track who did what, and when. Additionally, by using GitHub we can ensure that there is a history of the code as the project grows, allowing us to take steps back if needed. To ensure the quality of our work, we'll operate on a pull request/merge system; that is, when a branch is ready to be merged with the master branch, a pull request will be made and the code will be reviewed. Furthermore, if a goal isn't met, we will begin working together to meet the goal of that specific feature. To track what needs to be done, Github Issues will be made, and we can self-assign ourselves on what interests us, or what needs to be done. When the Issue is completed, it will be completed in the pull request section of GitHub.

To ensure goals are met the group will not move on to the next feature until the current feature is implemented fully, and doesn't cause conflicting problems with another feature.

3.4 Risk Management

Risk 1: Team Collaboration using Git may become a risk.
Likelihood of risk: Medium probability
Impact to project: Bad Github practice can lead to deletion or unwanted Modification in the project source code.
Address risks: To prevent this risks to occur, a good and clean work-flow structure need to be determined before any submission takes place and is expected that each developer uses its own branch to make changes to the root project.

Risk 2: Problems using Ruby on Rails
Likelihood of risk: Medium probability
Impact to project: If a good working comprehension of ruby is not accomplished, bugs and other undesirable outputs can affect the application functionality.
Address risk: To prevent risks related to bad ruby on Rails practice we will meet weekly to address the learning advances of each team member in relation of this language.

Risk 3: Scope Creep
Likelihood: Mid to hight
Impact to project: If new, not previously planned features are added to the application, we may have functionality problems, that can lead to the application not working or working incorrectly.
Address risk: To prevent non planned features to be added is important to follow a group work-flow system and report our advances each week.

4 Meeting Report

4.1 Progress Made

The team members working on the Zombie Apocalypse Prep App came together this week to introduce each other and get to know each person strengths and weaknesses in relationship to software engineering and programming in general.
We discussed how to approach the front and back end of the project, we also created a developing plan, and with a schedule to establish when is the most convenient time to meet for each member of the team. Additionally, we developed milestones to follow during the developing cycle, and established the goals for next week.

4.2 Next Week Goals

The general goal for next week is to create a baseline framework for the application. The framework will be pushed into our group github repository for further modifications.
The other goal is for the members that haven't used Ruby on Rails to download and become familiar with the software. The members will accomplish becoming familiar with the software by reading an online tutorial supplied by a on line textbook[1]. YouTube and similar platforms will also be used for extended research.

4.3 Contributions

All members working on the project contributed to the writing of this Latex document, Eric Sisson did the diagram that addresses how our approach plugs into existing resources, Kuan-Yu Lai made the use case diagram,, Christa Wright create the Schedule and Timeline section of the document and Jorge Guzman Nader wrote the meeting report and Project description section.
All members also help to set the scheduling of future meetings, the establishment of milestones for

the project and the goals to be accomplished in a given timeframe.
For future reference, the group will use Github as our main contribution platform, in which is expected of each member to work in a section of the code.

4.4 Customer

We meet with our costumer, and started to brain storm about the front-end phase of the web application, and how the main menu will look and act.
A diagram about tentative functionalities for the app was made, and some of the sub-field of the app were either introduced or discarded based on resources capabilities and time constrains.

5 Citations:

[1] https://www.railstutorial.org/book [?]
[2] https://www.w3schools.com/ [?]
[3] https://github.com/ [?]
[4] http://www.beyondorganic.net/dri-calculator.html [?]
[5] http://getbootstrap.com/ [?]