Final Project Report

The Student Game

Wesley Ackerman, Jacob Anderson, Matthew Hoiland, Kyle Richey

Table of Contents

Executive Summary

The goal of this project was to create an incremental game about the life of a college student using the MEAN stack. Although other games of this type exist, this project is unique because it uses user authentication to store and save the game state. This allows players to save the game at any point and return to it on any computer. This goal resulted in a complete and enjoyable game with potential for additional content. The application is hosted at http://cs360.kylerichey.com:3005/login.

Functional Description

The goal this application is to be a fun and addicting game. Games of this type are not meant to always be actively played. Instead it is intended that it runs in the background for some interval of time. When the player returns to the game, he can quickly spend the accumulated resources and queue up new activities for the next time. It is intended that the player is able to spend as much time as he wants in the game during any given session, while still enjoying an engaging experience. This is accomplished by creating short and long term activities within the game. Some resources accumulate quickly and cap out, while others generate slowly without any limit.

This results in a meaningful experience every time you log in, whether it be for 30 seconds or an hour.

The various use cases are explained below where each page or tab that the user can see are outlined. The original plan was to include a hobbies section where miscellaneous activities would go. This included a “Ramen” system where the player would feed his character a certain amount every day or he would die. Ultimately this system was not implemented because it didn’t result in engaging gameplay, but became a chore to complete on a regular basis. Instead the idea was expanded into new sections called Relationships and Gym.

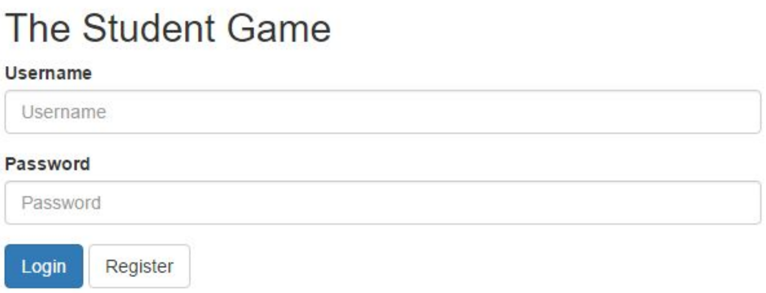
Methods

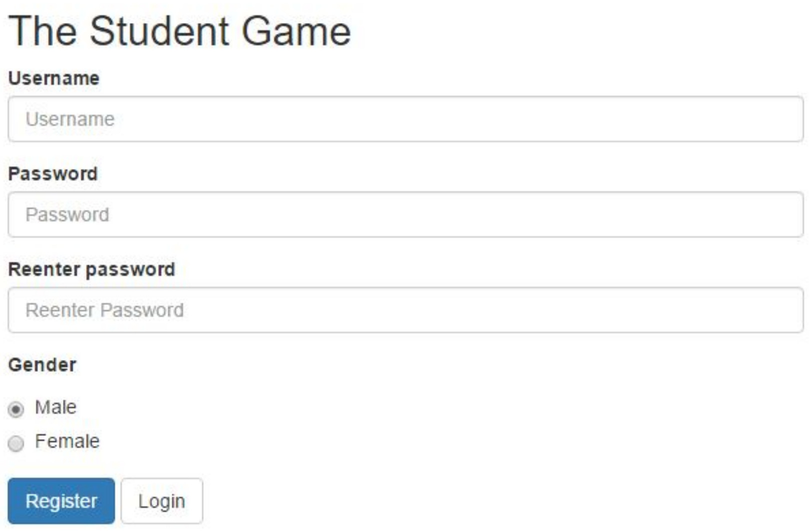
Each member of the group had a specific emphasis when working on the project. Kyle focused on the frontend, creating the dynamically generated content that the user sees. Matt developed the authentication and registration system that the application uses. Wes worked on saving and loading the user state in MongoDB. Jacob helped in developing the frontend behavior, and also handled most of the write-ups for our group. All members worked together in designing gameplay.

Login and Registration

This page serves as the entry point into the application. Registration is required to play the game. The user creates an account and then may enter the game. This ensures that when the game autosaves the game state is stored and connected to the correct user. This allows the user to play the game on any computer.

The login and registration views have been refined. Any errors are rendered to the page using EJS. The page validates that a username exists, passwords match, and that after registration, that username is unique. It also validates that passwords are not empty.

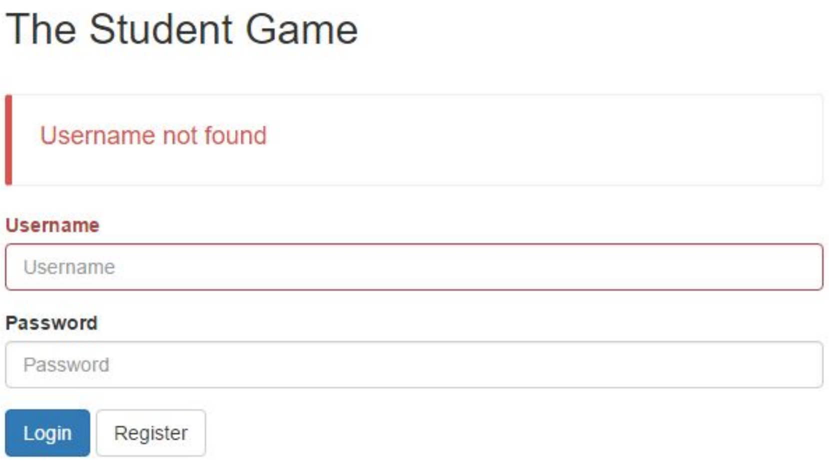




The Student Game

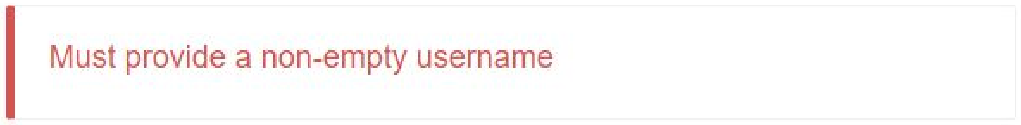
● Male

○ Female

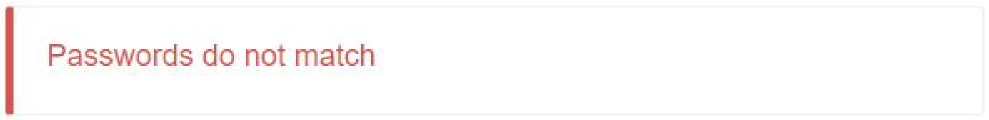


The Student Game

Username not found



Must provide a non-empty username

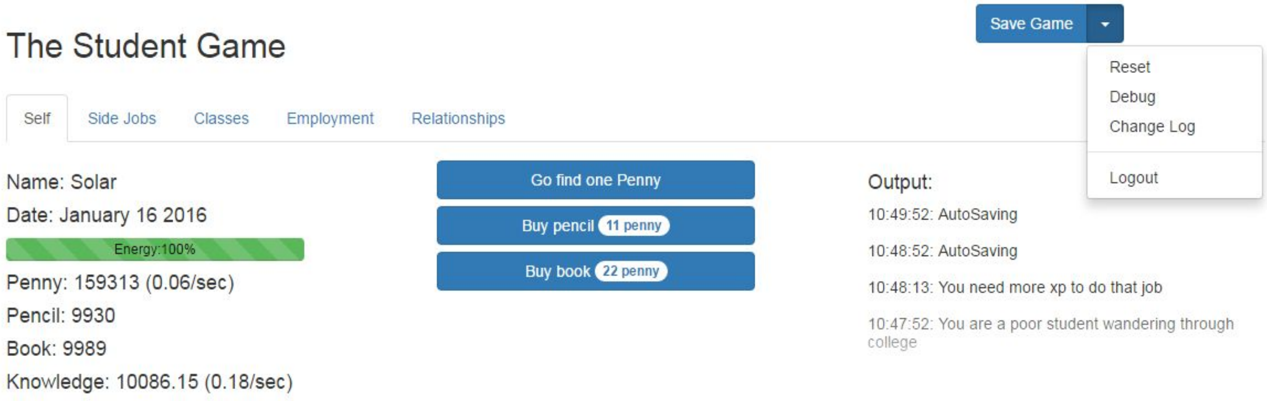


Passwords do not match

Self

The Self tab is the primary tab of the Student Game. In the center column, the buttons on display a single unique resource. An item's cost will increase based on the quantity already purchased of that item. The number on the button updates to reflect the current cost. In the left column, the primary game data is displayed. This includes the player's name, the current date, current energy level, and the current amount of various resources, with their respective automatically generating rates. In the rightmost column, the game displays important information, error messages, and progress.

In the top right corner is the Save Game button. Pressing this button stores the entire state of the game on the Mongo database. This allows the player to resume play where they left off. The game autosaves every minute, and notifies the user in the Output pane. Next to the save button there is a drop down tab providing basic options. These include a reset button that clears the entire game, a debug button that provides immediate access to all of the game's content and a large amount of resources, and a change log which shows the development roadmap. Last is the logout button which saves the game and returns the user to the login screen.



The Student Game

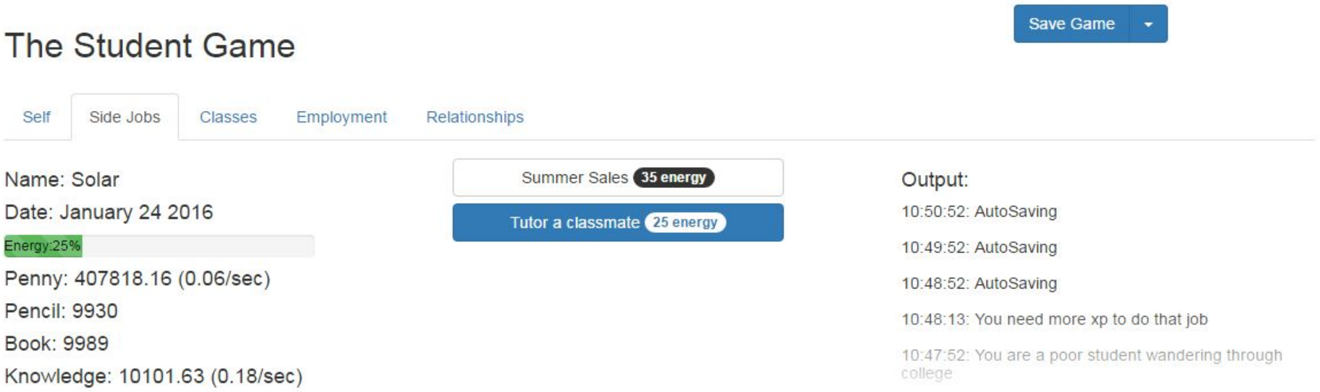
Name: Solar
Date: January 16 2016
Energy:100%
Penny: 159313 (0.06/sec)
Pencil: 9930
Book: 9989
Knowledge: 10086.15 (0.18/sec)

Output:

```
10:49:52: AutoSaving
10:48:52: AutoSaving
10:48:13: You need more xp to do that job
10:47:52: You are a poor student wandering through college
```

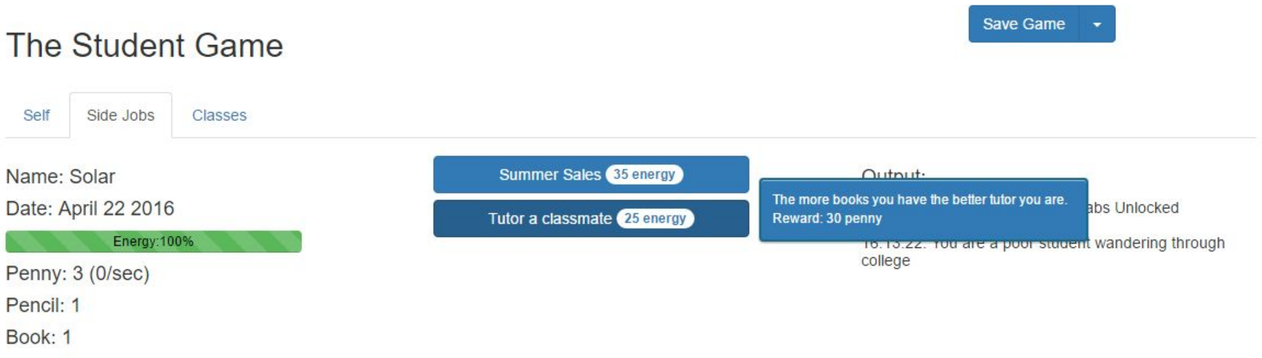
Side Jobs

New tabs are unlocked based on gameplay. Discovering which activities unlock a new tab is part of the game. Side Jobs are opened when the player buys their first book. Side jobs use energy, a resource which refills at a fixed rate and cannot be upgraded. Like every button in the game, the specific job will switch between active or disabled based on current availability of the required resource. In the screencap below, the Summer Sales job is not illuminated because the player only has 25 energy at that moment and it requires 35. However the Tutor job only requires 25 energy so it is available to click. Jobs trade player energy for pennies. The reward from a job scales based on the current amount of resources. The total reward is automatically calculated and displayed in the hover over tooltip.



Name: Solar
Date: January 24 2016
Penny: 407818.16 (0.06/sec)
Pencil: 9930
Book: 9989
Knowledge: 10101.63 (0.18/sec)

Output:
10:50:52: AutoSaving
10:49:52: AutoSaving
10:48:52: AutoSaving
10:48:13: You need more xp to do that job
10:47:52: You are a poor student wandering through college



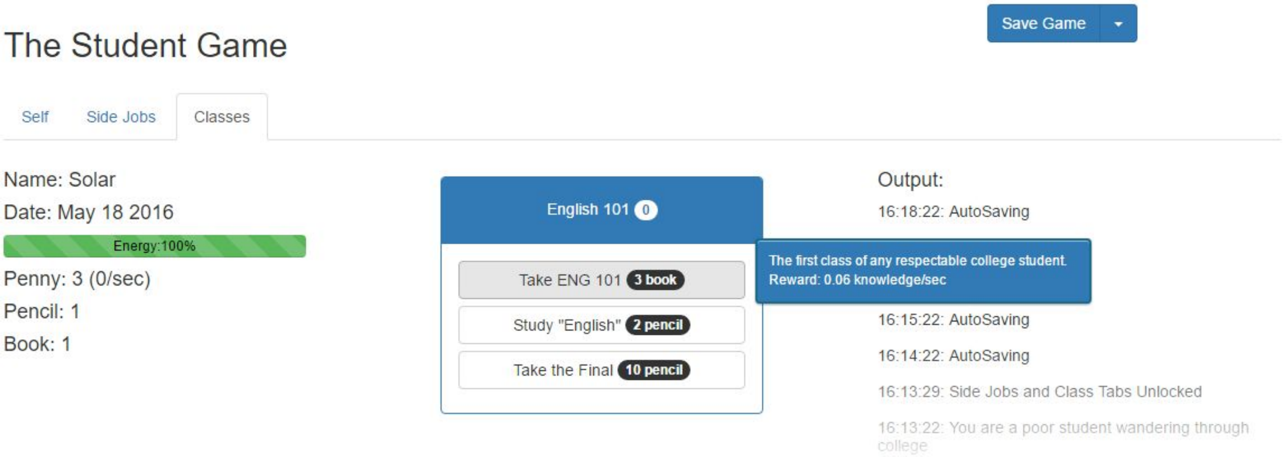
Name: Solar
Date: April 22 2016
Penny: 3 (0/sec)
Pencil: 1
Book: 1

Output:
abs Unlocked
16:13:22: You are a poor student wandering through college

The more books you have the better tutor you are.
Reward: 30 penny

Classes

The Classes tab is opened upon purchase of the player's first book. Taking a class requires a certain amount of books. It also unlocks a new resource called knowledge and grants a small amount over time. Studying requires pencils and grants a fixed amount of knowledge. Taking a class's final grants a fixed amount of knowledge and unlocks new classes.

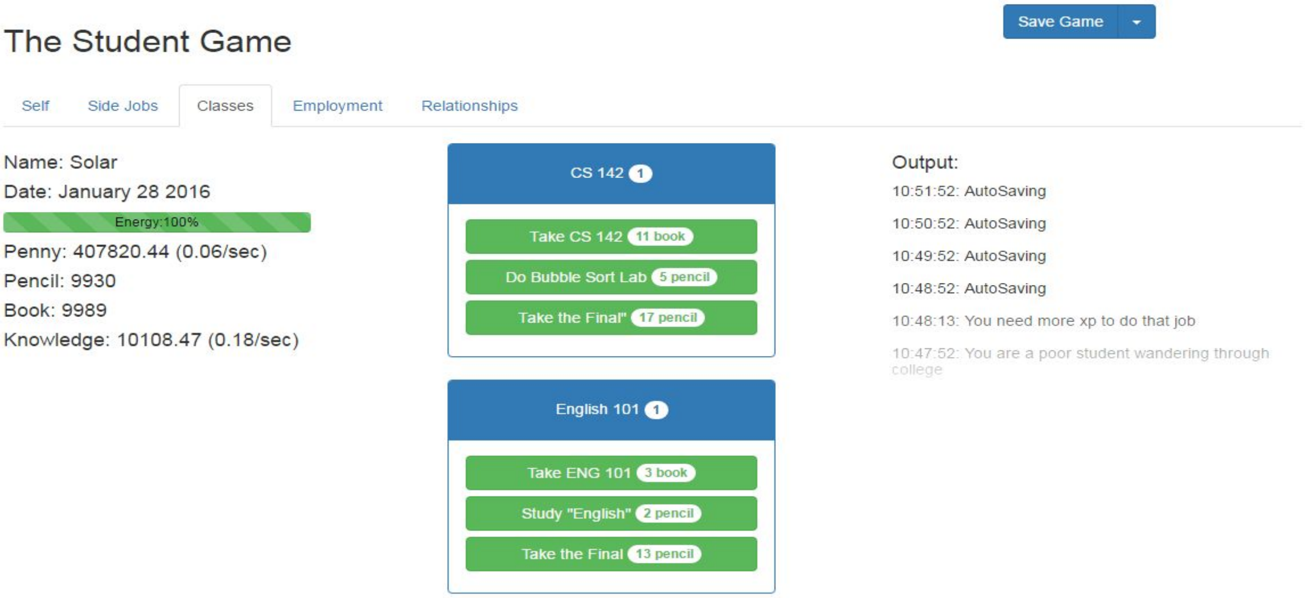


Name: Solar
Date: May 18 2016
Penny: 3 (0/sec)
Pencil: 1
Book: 1

The first class of any respectable college student.
Reward: 0.06 knowledge/sec

Output:

```
16:18:22: AutoSaving
16:15:22: AutoSaving
16:14:22: AutoSaving
16:13:29: Side Jobs and Class Tabs Unlocked
16:13:22: You are a poor student wandering through
college
```



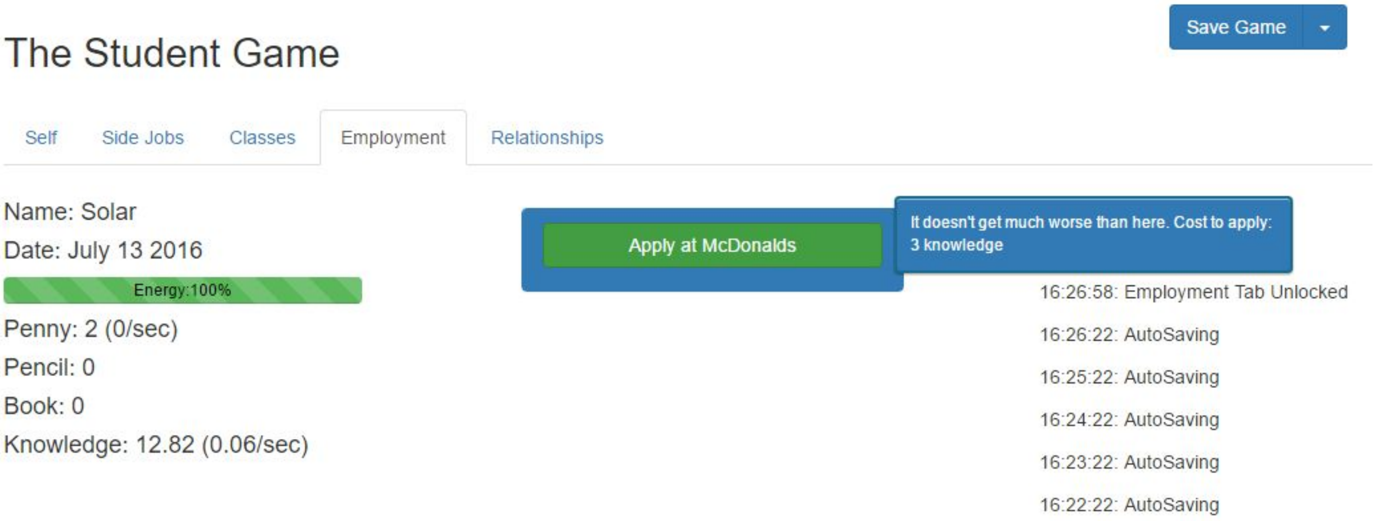
Name: Solar
Date: January 28 2016
Penny: 407820.44 (0.06/sec)
Pencil: 9930
Book: 9989
Knowledge: 10108.47 (0.18/sec)

Output:

```
10:51:52: AutoSaving
10:50:52: AutoSaving
10:49:52: AutoSaving
10:48:52: AutoSaving
10:48:13: You need more xp to do that job
10:47:52: You are a poor student wandering through
college
```

Employment

The Employment tab is unlocked when the player takes their first class. The player applies for a job by using their knowledge. After applying at a place of employment, the player chooses a position there. Once the player has chosen a position, an experience meter begins filling up, unlocking new jobs that will provide more pennies over time. New jobs are opened up by taking new classes.



Name: Solar

Date: July 13 2016

Penny: 2 (0/sec)

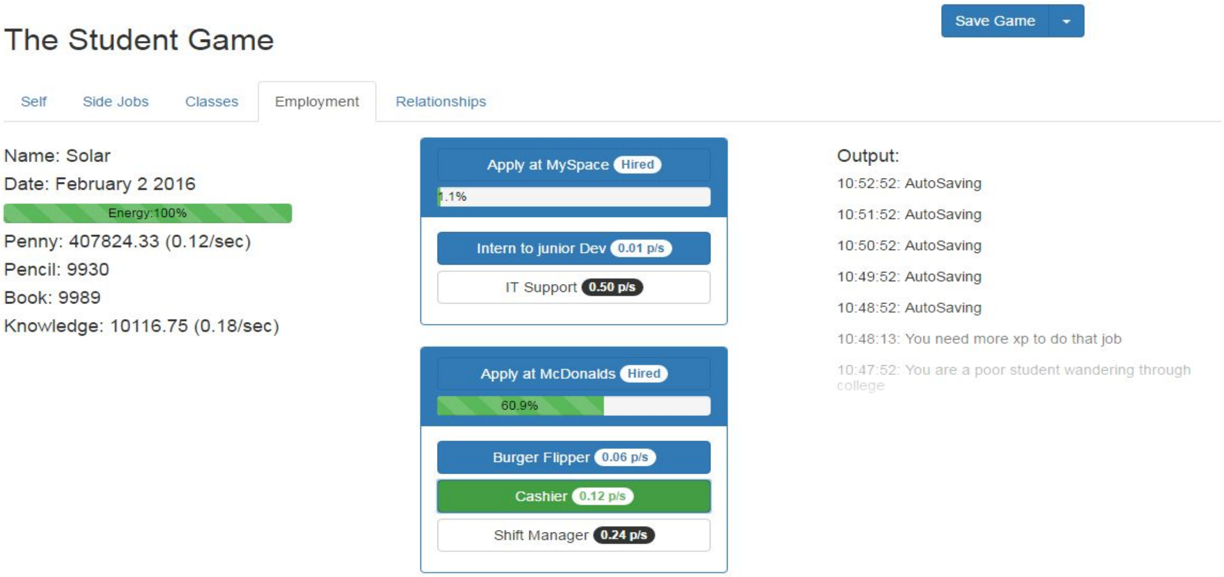
Pencil: 0

Book: 0

Knowledge: 12.82 (0.06/sec)

It doesn't get much worse than here. Cost to apply: 3 knowledge

```
16:26:58: Employment Tab Unlocked
16:26:22: AutoSaving
16:25:22: AutoSaving
16:24:22: AutoSaving
16:23:22: AutoSaving
16:22:22: AutoSaving
```



Name: Solar

Date: February 2 2016

Penny: 407824.33 (0.12/sec)

Pencil: 9930

Book: 9989

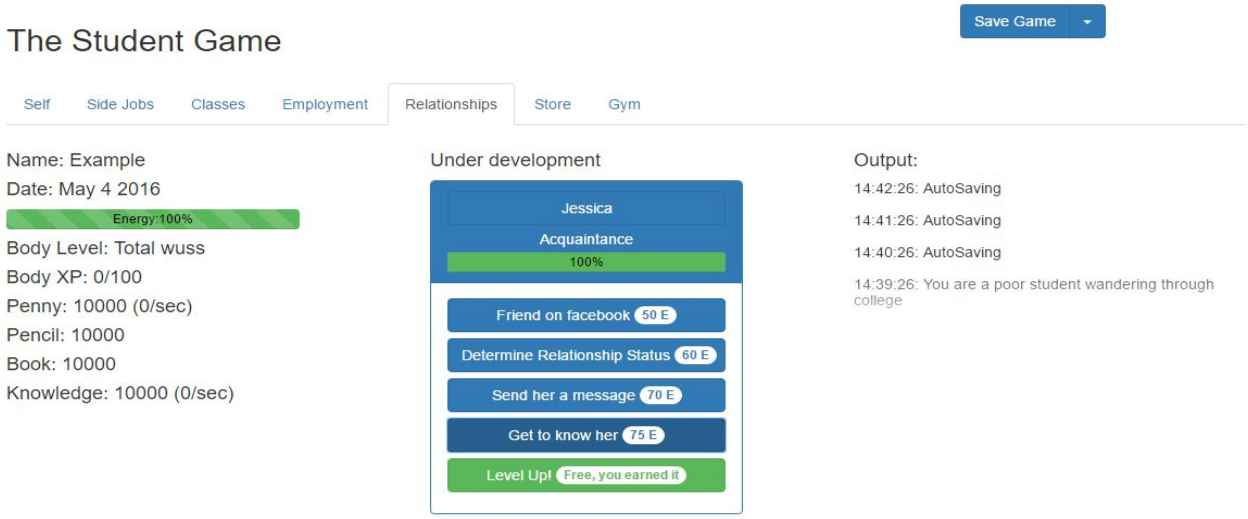
Knowledge: 10116.75 (0.18/sec)

Output:

```
10:52:52: AutoSaving
10:51:52: AutoSaving
10:50:52: AutoSaving
10:49:52: AutoSaving
10:48:52: AutoSaving
10:48:13: You need more xp to do that job
10:47:52: You are a poor student wandering through
college
```

Relationships

Relationships become available after taking a class. The game randomly generates a person to date with a corresponding tooltip description from a list. After initial contact, the player has to complete various tasks to move from Acquaintance, to New Friend, to Girlfriend. The tasks begin by only consuming energy, but after leveling up to New Friend the tasks become less defined. The player can choose which activities to complete to gain more experience. The player may choose give the lucky girl gifts of flowers and chocolate, or hang out with her. Each option requires different resources and increase a progress bar to the next level by a different amount. Eventually the player can ask her out and she becomes your girlfriend. Currently, the relationship tab is only fully implemented if a user is registered as male.



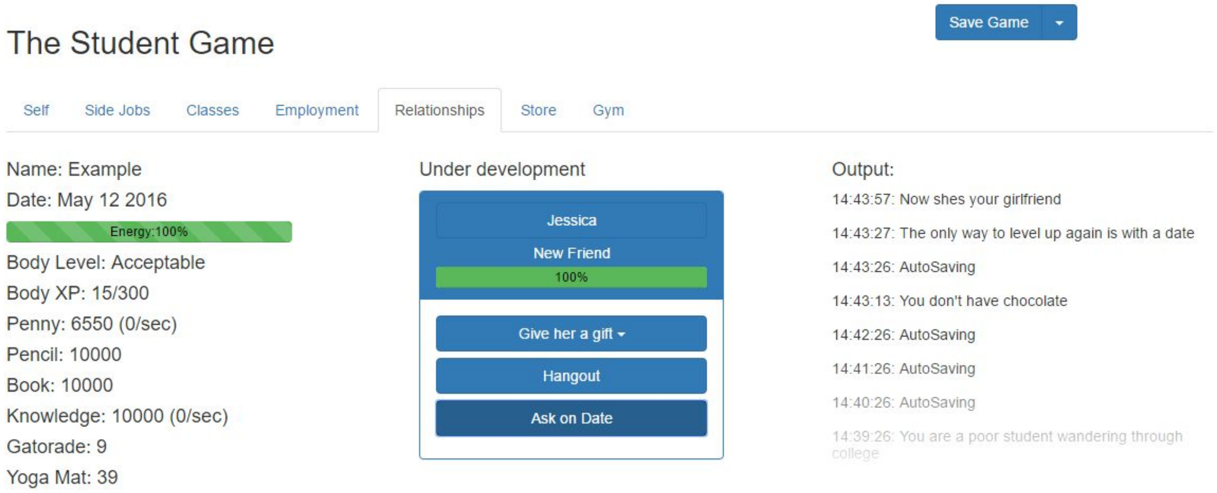
The Student Game

Name: Example
Date: May 4 2016
Body Level: Total wuss
Body XP: 0/100
Penny: 10000 (0/sec)
Pencil: 10000
Book: 10000
Knowledge: 10000 (0/sec)

Under development

Output:

```
14:42:26: AutoSaving
14:41:26: AutoSaving
14:40:26: AutoSaving
14:39:26: You are a poor student wandering through
college
```



The Student Game

Name: Example
Date: May 12 2016
Body Level: Acceptable
Body XP: 15/300
Penny: 6550 (0/sec)
Pencil: 10000
Book: 10000
Knowledge: 10000 (0/sec)
Gatorade: 9
Yoga Mat: 39

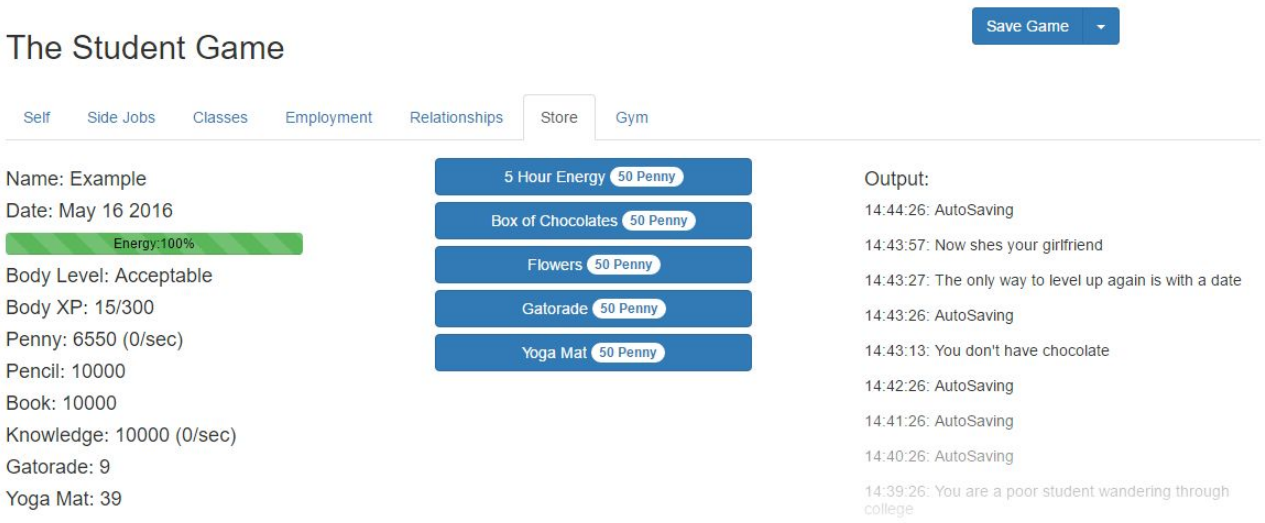
Under development

Output:

```
14:43:57: Now shes your girlfriend
14:43:27: The only way to level up again is with a date
14:43:26: AutoSaving
14:43:13: You don't have chocolate
14:42:26: AutoSaving
14:41:26: AutoSaving
14:40:26: AutoSaving
14:39:26: You are a poor student wandering through
college
```

Store

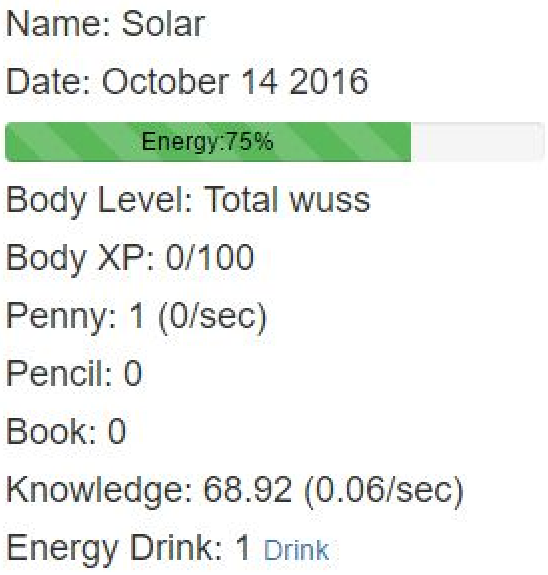
The Store tab allows the player to purchase resources in areas other than education. The Store is unlocked when the player first attempts to find engage in a relationship. The Store has various items such as Gatorade or Yoga Mat, which are resources for the gym, 5 Hour Energy, which instantly replenishes the players energy to 100%, Box of Chocolates or Flowers which can be given as gifts in the relationship tab.



Name: Example
Date: May 16 2016
Body Level: Acceptable
Body XP: 15/300
Penny: 6550 (0/sec)
Pencil: 10000
Book: 10000
Knowledge: 10000 (0/sec)
Gatorade: 9
Yoga Mat: 39

Output:

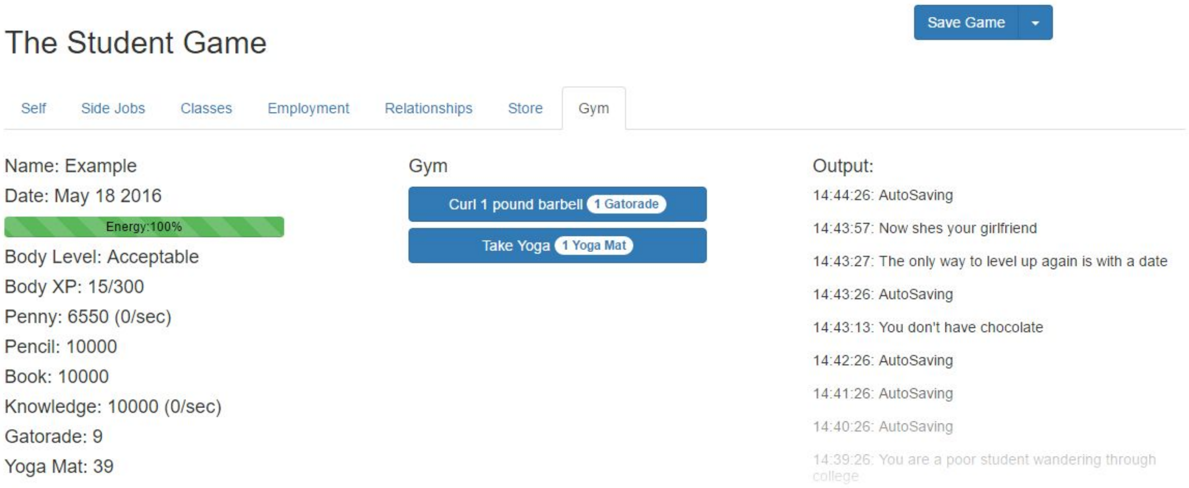
```
14:44:26: AutoSaving
14:43:57: Now shes your girlfriend
14:43:27: The only way to level up again is with a date
14:43:26: AutoSaving
14:43:13: You don't have chocolate
14:42:26: AutoSaving
14:41:26: AutoSaving
14:40:26: AutoSaving
14:39:26: You are a poor student wandering through college
```



Name: Solar
Date: October 14 2016
Body Level: Total wuss
Body XP: 0/100
Penny: 1 (0/sec)
Pencil: 0
Book: 0
Knowledge: 68.92 (0.06/sec)
Energy Drink: 1

Gym

The Gym is unlocked when the player first attempts to find engage in a relationship. The gym requires resources bought from the store, which are used to level up the player’s Body Type. The scale begins at Total Wuss, then progresses to Weak, and then to Acceptable. When the player’s Body Type reaches acceptable, they may gain a girlfriend after filling up the respective progress bar in the Relationships tab.



Name: Example
Date: May 18 2016
Body Level: Acceptable
Body XP: 15/300
Penny: 6550 (0/sec)
Pencil: 10000
Book: 10000
Knowledge: 10000 (0/sec)
Gatorade: 9
Yoga Mat: 39

```
14:44:26: AutoSaving
14:43:57: Now shes your girlfriend
14:43:27: The only way to level up again is with a date
14:43:26: AutoSaving
14:43:13: You don't have chocolate
14:42:26: AutoSaving
14:41:26: AutoSaving
14:40:26: AutoSaving
14:39:26: You are a poor student wandering through
college
```

Conclusion

We learned that creating a web application doesn’t not have to be a complex effort. Through the aid of Mongo, Angular, Express, and Node the initial setup was straightforward. The applications worked together very well. This created an environment with the tools necessary to complete the incremental game as it was envisioned.

If we were to do the project again, it may have been beneficial to map out the more details of the application before writing any code. This would have made it easier to equally divide the work so that each member group had a more complete idea of their responsibilities. The informal approach of coding often resulted in minor code and design collisions that could have potentially been avoided.

None of the technologies used could be considered disappointments. The MEAN stack was incredibly valuable. The frontend was implemented with HTML and Bootstrap. This was very useful in providing a good UI. The backend code was all javascript, accessed using Angular. The source is found here https://github.com/kylerichey/CS360FinalProject.