Juhyun Oh, Kevin Kim, Zhisheng Liu

CSE134B HW3

In this homework, our group implemented styling HTML templates with two different methods. The first method, Vanilla, used no other pre-configured frameworks or outside imports. Instead we manually tested each stylesheet per HTML file. For the second method, we used a well-known framework for building websites, Bootstrap. By implementing our web app with two differents methods we were able to understand the tradeoffs of using one method to another.

With the Vanilla method, we had the freedom to style our HTML elements in whichever way we wanted. Thus we had to first consider the tradeoffs of usability and style in advance. In contrast, Bootstrap forced our styling into a well established framework. We did not have to reinvent the wheel.

The time to implement Bootstrap took about 1.5 hours per page and for Vanilla we took about 3 hours per page. Because this was our first time writing CSS, the freedom we got from Vanilla translated into a considerable amount of time in debugging and testing.However, this may have led us to become better and faster when it came time to implement our app with the Bootstrap method. Looking at the statistics, however, we wrote significantly less CSS lines for our Bootstrapped HTML file.

Although we spent less time for the Bootstrap method, there were many problems in trying to change the given framework to suit our needs. We quickly realized that it was difficult to make changes to the given framework. This made us less flexible in what we could implement.

A huge advantage of Bootstrap was the grid system. Because the grid system already considers different size screens and collapses them upon changing screen sizes we did not have to spend extra time in implementing different styles for different platforms. For the vanilla method, we only styled for the Desktop and only changed the scaling. Furthermore, the scaling problem (for example, overflow while resizing browser) in vanilla styling keep happening if we do not use a table to hold content, but using table restrict some presentational utilities and freedom. One more advantage of Bootstrap over vanilla styling is that it provides great consistency of the design.

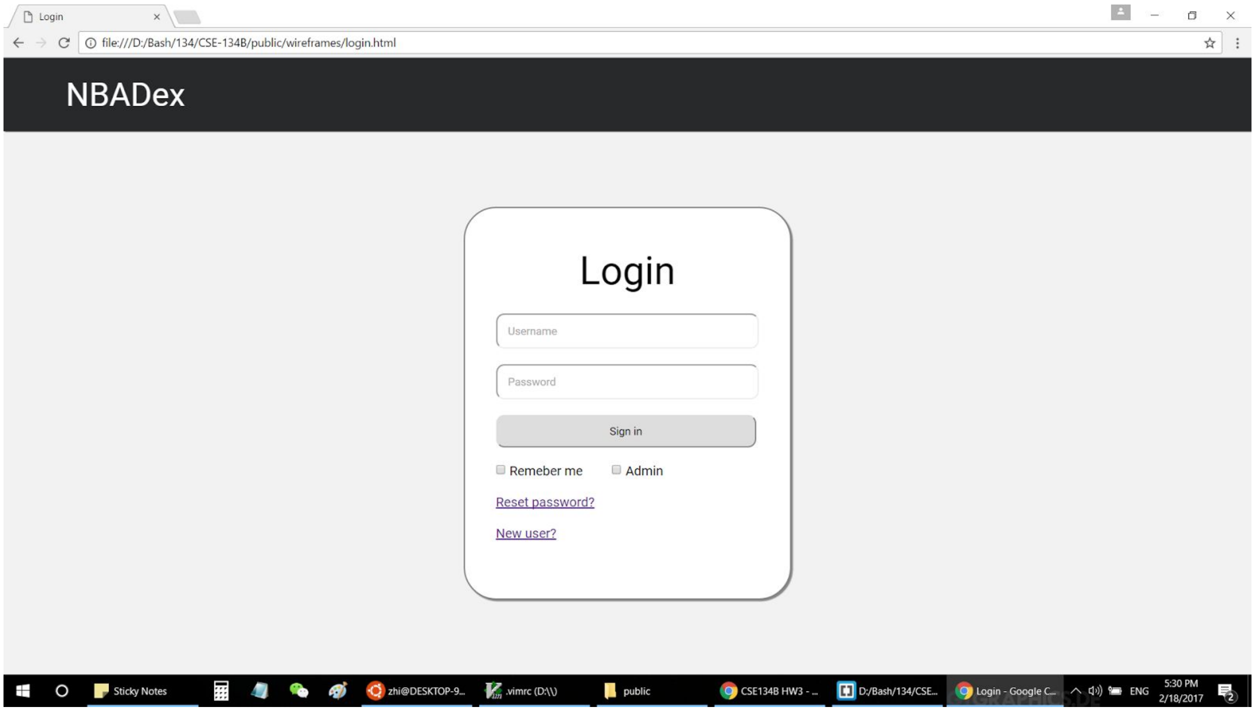
In aspect of performance, we tried using different methods for Bootstrap. We downloaded a package from Bootstrap which contained the files we needed for our HTML elements and hosted them. The other method, we linked a given CDN for Bootstrap for each of our HTML pages. For the vanilla method, we wrote a separate CSS file for each HTML file and hosted each file. We found that the bootstrap.min.css file was about 78KB. However the size of our bootstrap file was dwarfed by the bytes of each NBA players' pictures. The load time and rendering however, took .8 seconds longer.

Overall, because of the rise of mobile and tablet browsers it may be much easier for the developer to use Bootstrap or another pre-configured framework. However we know that the load times will be slower and we will be pigeon-holed into creating our web app in a certain but proper way. For the Vanilla method, we have much more freedom and we can optimize our load time. However if we do not have experience with UI then we can end up spending much more time and unintentionally create web apps that are not user friendly.

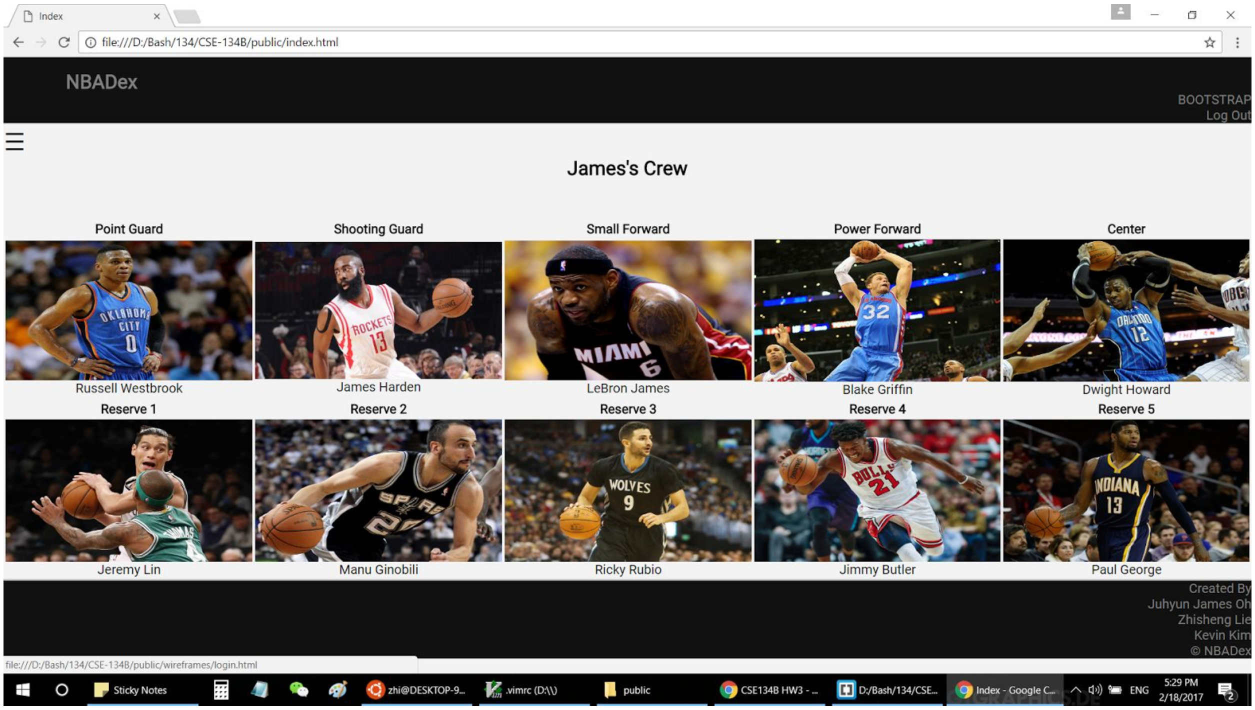
	Vanilla	Bootstrap
SLOC (html)	139 lines	124 lines
SLOC (css)	113 lines	30 lines
Byte Count	1.4MB	1.4MB
Time to load w/o caching	2s	2.88s
Hours Spent	12 hours	6 hours

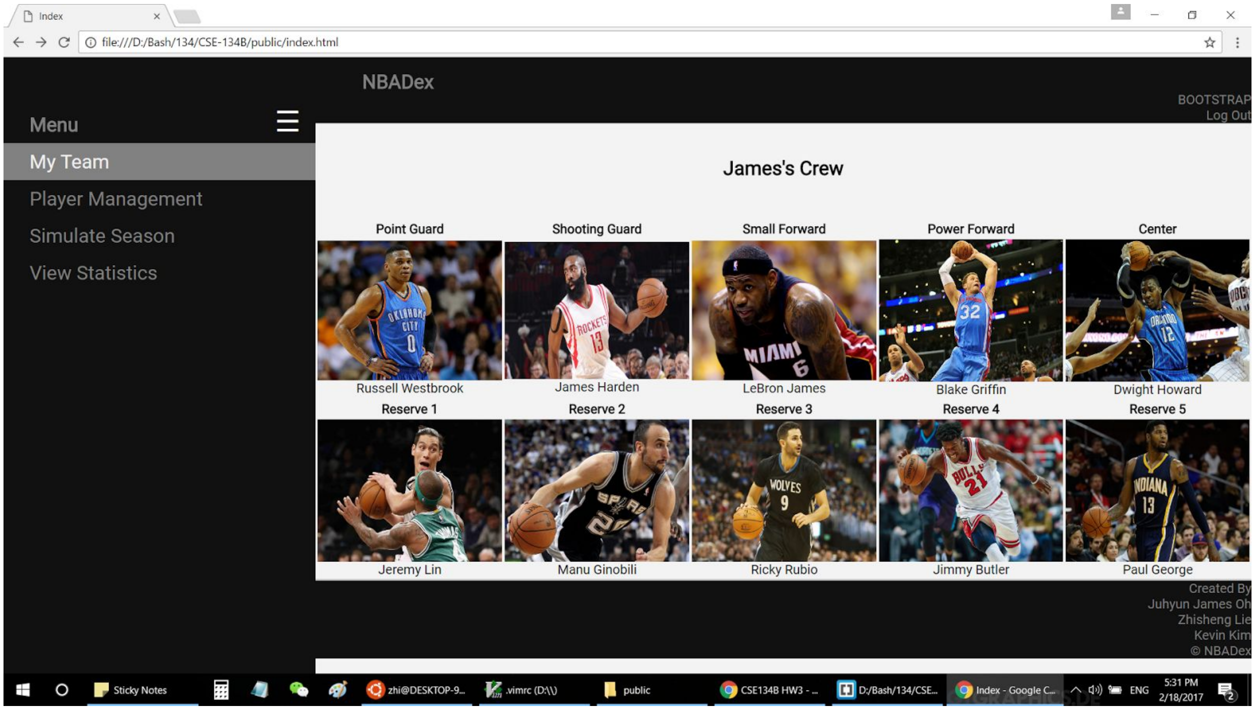
Execution screens of Vanilla Styling version:

1. login screen

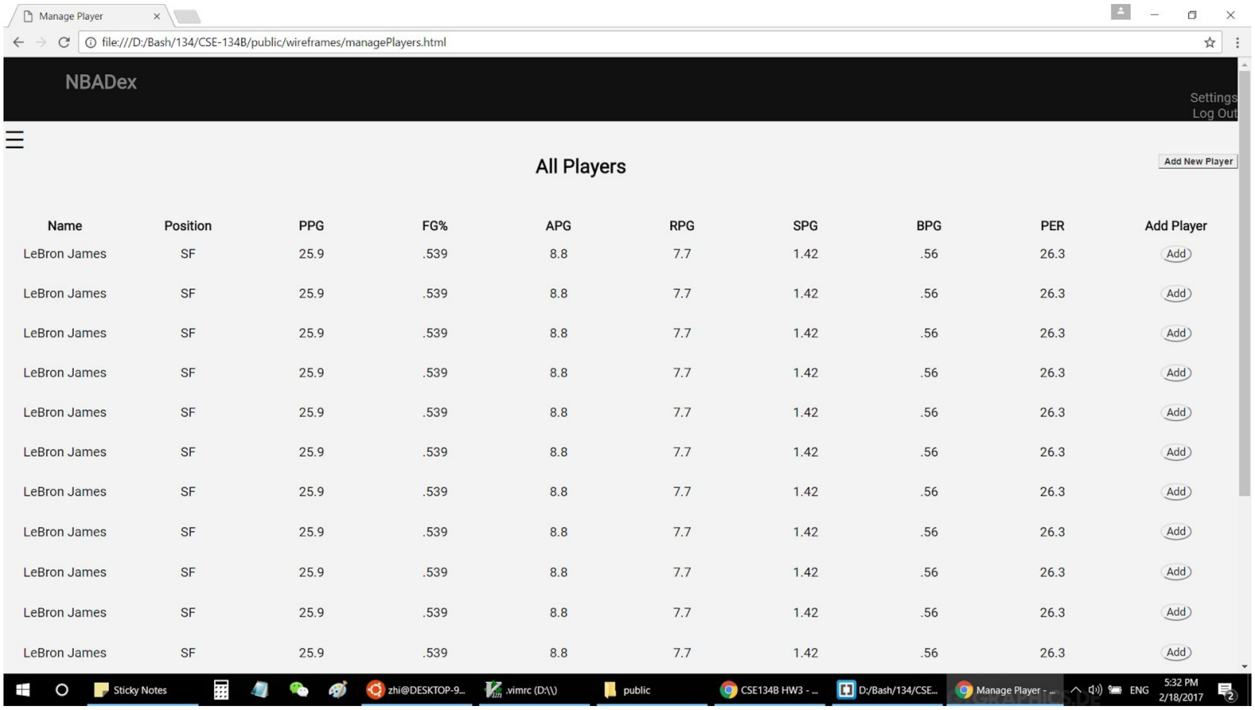


2. main page (My Team page)





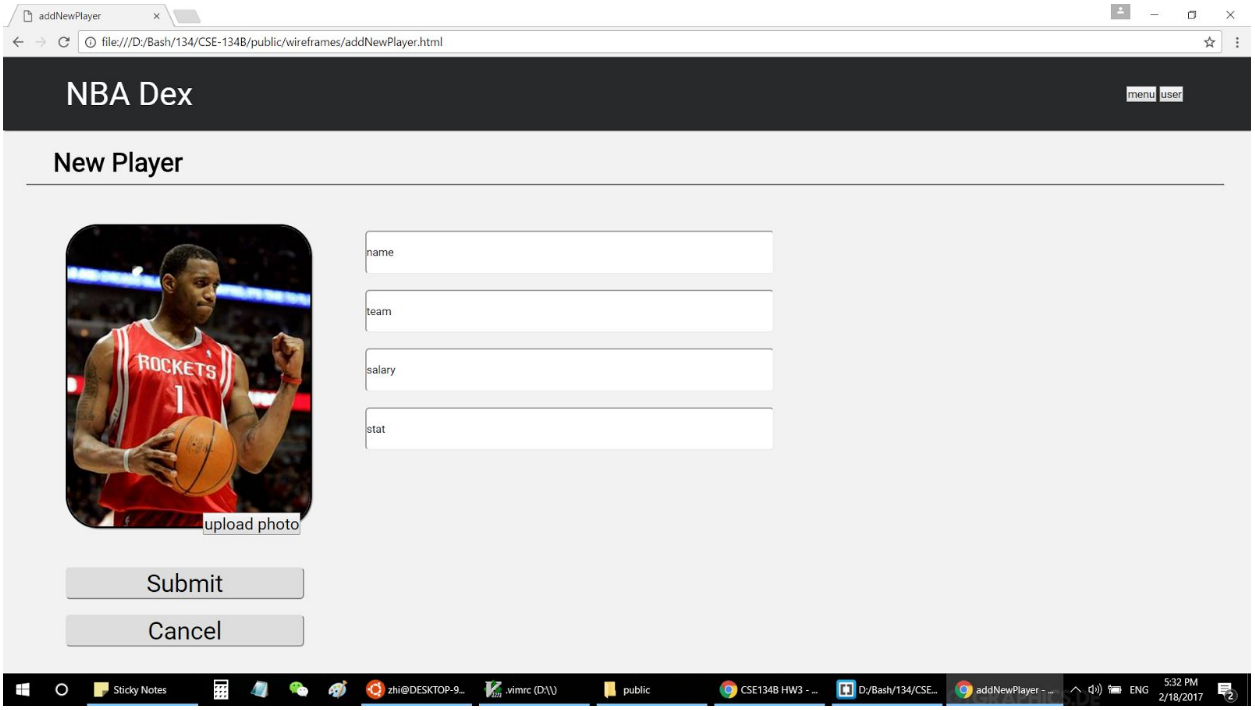
3. Player Management



All Players

Name	Position	PPG	FG%	APG	RPG	SPG	BPG	PER	Add Player
LeBron James	SF	25.9	.539	8.8	7.7	1.42	.56	26.3	Add
LeBron James	SF	25.9	.539	8.8	7.7	1.42	.56	26.3	Add
LeBron James	SF	25.9	.539	8.8	7.7	1.42	.56	26.3	Add
LeBron James	SF	25.9	.539	8.8	7.7	1.42	.56	26.3	Add
LeBron James	SF	25.9	.539	8.8	7.7	1.42	.56	26.3	Add
LeBron James	SF	25.9	.539	8.8	7.7	1.42	.56	26.3	Add
LeBron James	SF	25.9	.539	8.8	7.7	1.42	.56	26.3	Add
LeBron James	SF	25.9	.539	8.8	7.7	1.42	.56	26.3	Add
LeBron James	SF	25.9	.539	8.8	7.7	1.42	.56	26.3	Add
LeBron James	SF	25.9	.539	8.8	7.7	1.42	.56	26.3	Add
LeBron James	SF	25.9	.539	8.8	7.7	1.42	.56	26.3	Add

4. Add New Player



New Player

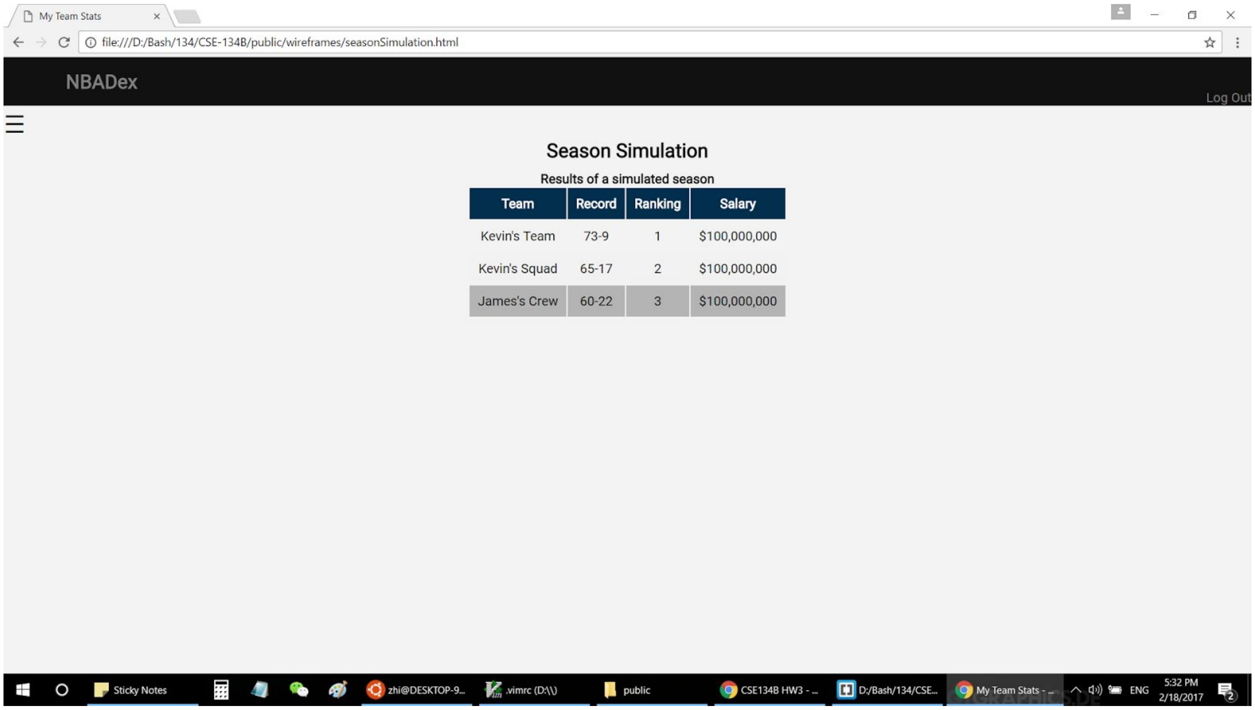
name

team

salary

stat

5. Simulation Page

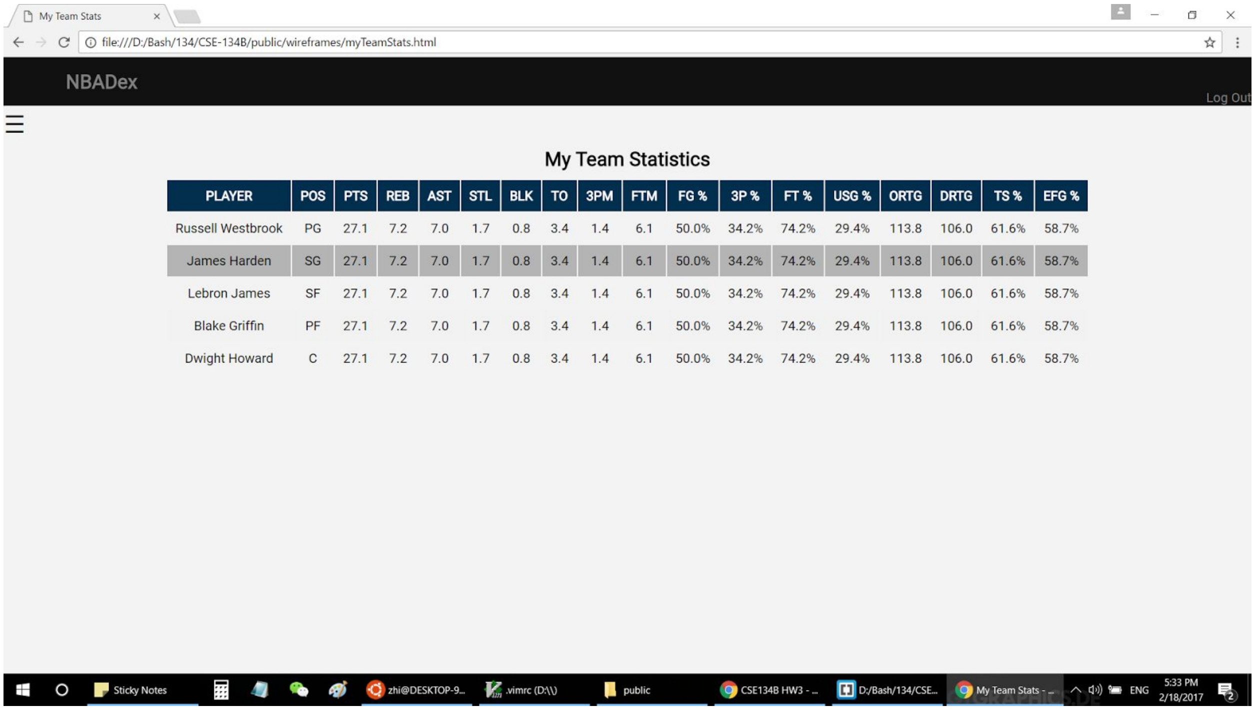


Season Simulation

Results of a simulated season

Team	Record	Ranking	Salary
Kevin's Team	73-9	1	$100,000,000
Kevin's Squad	65-17	2	$100,000,000
James's Crew	60-22	3	$100,000,000

6. Statistic page

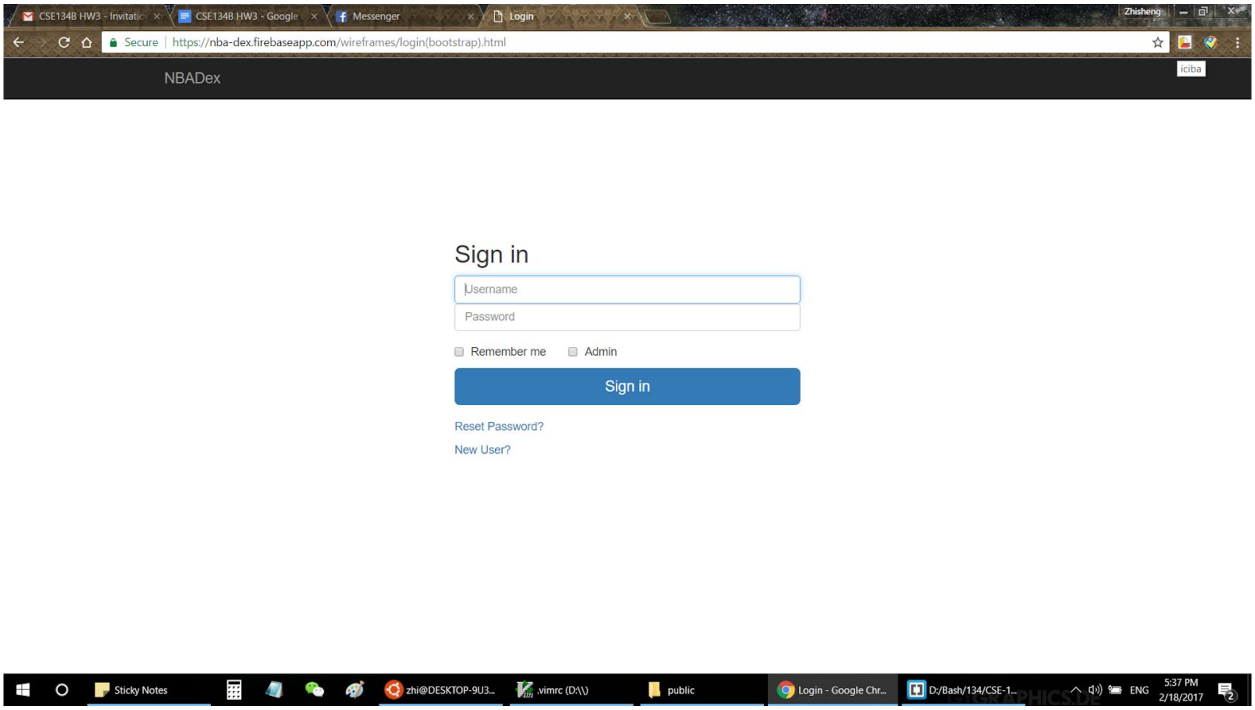


My Team Statistics

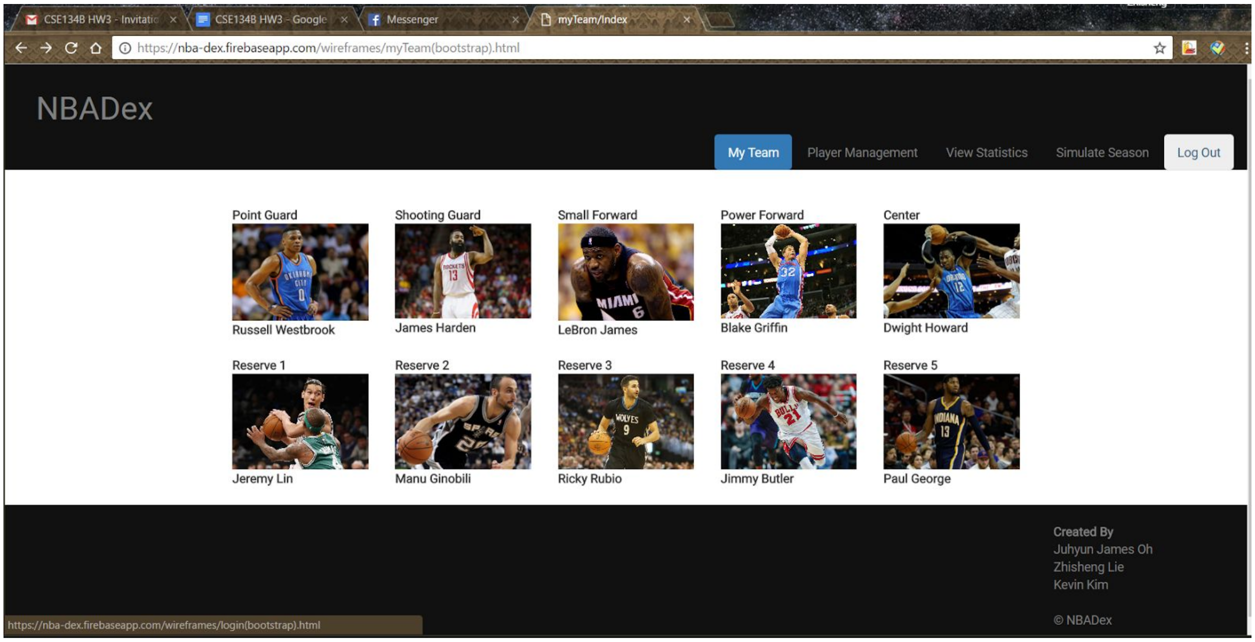
PLAYER	POS	PTS	REB	AST	STL	BLK	TO	3PM	FTM	FG %	3P %	FT %	USG %	ORTG	DRTG	TS %	EFG %
Russell Westbrook	PG	27.1	7.2	7.0	1.7	0.8	3.4	1.4	6.1	50.0%	34.2%	74.2%	29.4%	113.8	106.0	61.6%	58.7%
James Harden	SG	27.1	7.2	7.0	1.7	0.8	3.4	1.4	6.1	50.0%	34.2%	74.2%	29.4%	113.8	106.0	61.6%	58.7%
Lebron James	SF	27.1	7.2	7.0	1.7	0.8	3.4	1.4	6.1	50.0%	34.2%	74.2%	29.4%	113.8	106.0	61.6%	58.7%
Blake Griffin	PF	27.1	7.2	7.0	1.7	0.8	3.4	1.4	6.1	50.0%	34.2%	74.2%	29.4%	113.8	106.0	61.6%	58.7%
Dwight Howard	C	27.1	7.2	7.0	1.7	0.8	3.4	1.4	6.1	50.0%	34.2%	74.2%	29.4%	113.8	106.0	61.6%	58.7%

Execution screens of <u>Bootstrap Styling</u> version:

1. login screen



2. main page (My Team page)



Point Guard
Russell Westbrook

Shooting Guard
James Harden

Small Forward
LeBron James

Power Forward
Blake Griffin

Center
Dwight Howard

Reserve 1
Jeremy Lin

Reserve 2
Manu Ginobili

Reserve 3
Ricky Rubio

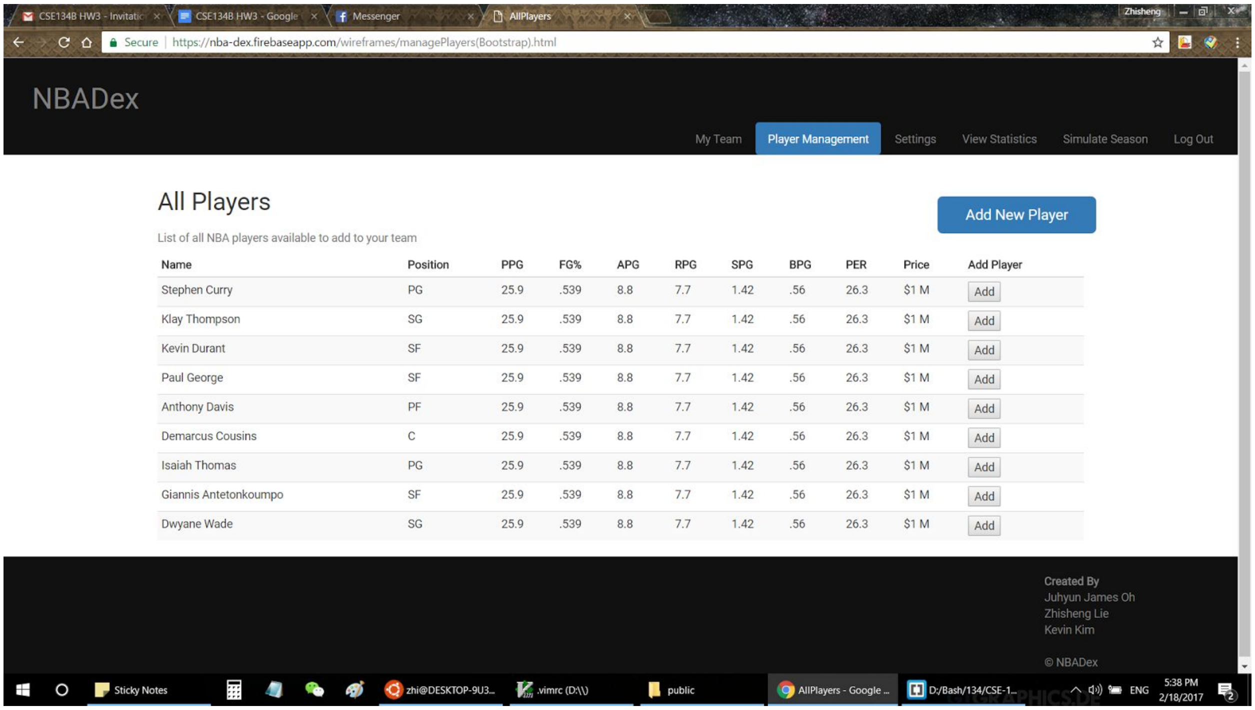
Reserve 4
Jimmy Butler

Reserve 5
Paul George

Created By
Juhyun James Oh
Zhisheng Lie
Kevin Kim

© NBADex

3. Player Management



All Players

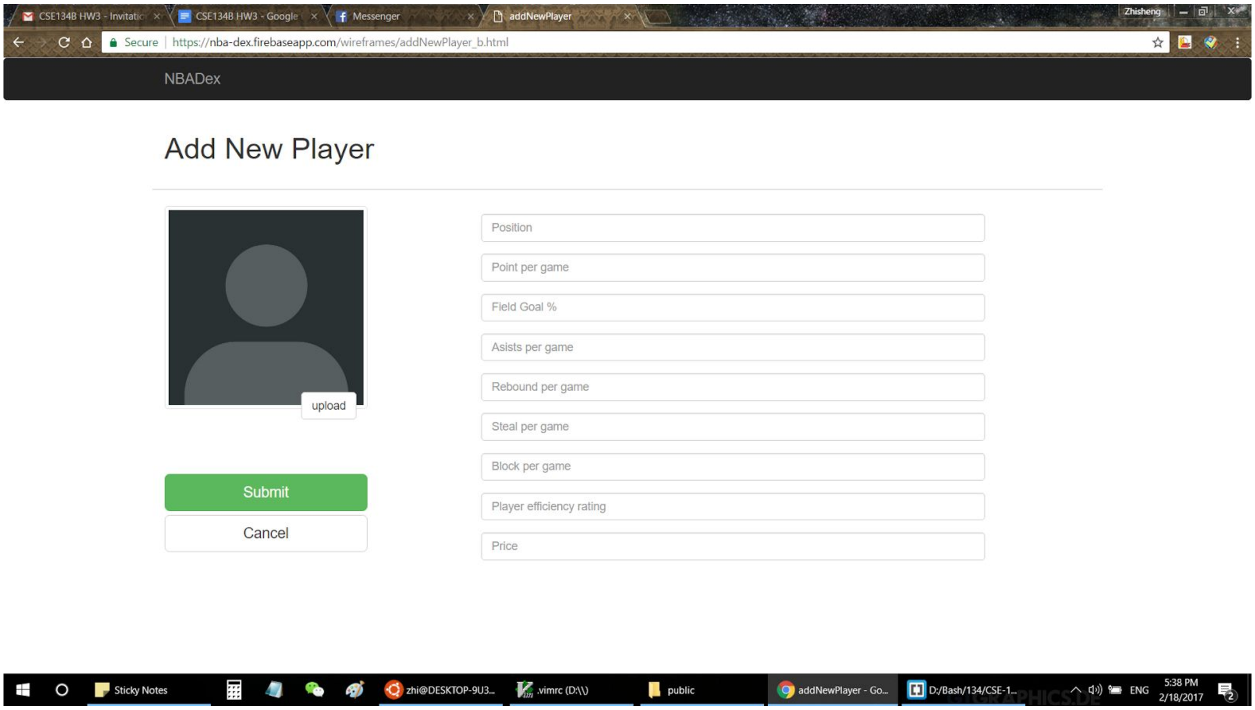
List of all NBA players available to add to your team

Name	Position	PPG	FG%	APG	RPG	SPG	BPG	PER	Price	Add Player
Stephen Curry	PG	25.9	.539	8.8	7.7	1.42	.56	26.3	$1 M	Add
Klay Thompson	SG	25.9	.539	8.8	7.7	1.42	.56	26.3	$1 M	Add
Kevin Durant	SF	25.9	.539	8.8	7.7	1.42	.56	26.3	$1 M	Add
Paul George	SF	25.9	.539	8.8	7.7	1.42	.56	26.3	$1 M	Add
Anthony Davis	PF	25.9	.539	8.8	7.7	1.42	.56	26.3	$1 M	Add
Demarcus Cousins	C	25.9	.539	8.8	7.7	1.42	.56	26.3	$1 M	Add
Isaiah Thomas	PG	25.9	.539	8.8	7.7	1.42	.56	26.3	$1 M	Add
Giannis Antetonkoumpo	SF	25.9	.539	8.8	7.7	1.42	.56	26.3	$1 M	Add
Dwyane Wade	SG	25.9	.539	8.8	7.7	1.42	.56	26.3	$1 M	Add

Created By
Juhyun James Oh
Zhisheng Lie
Kevin Kim

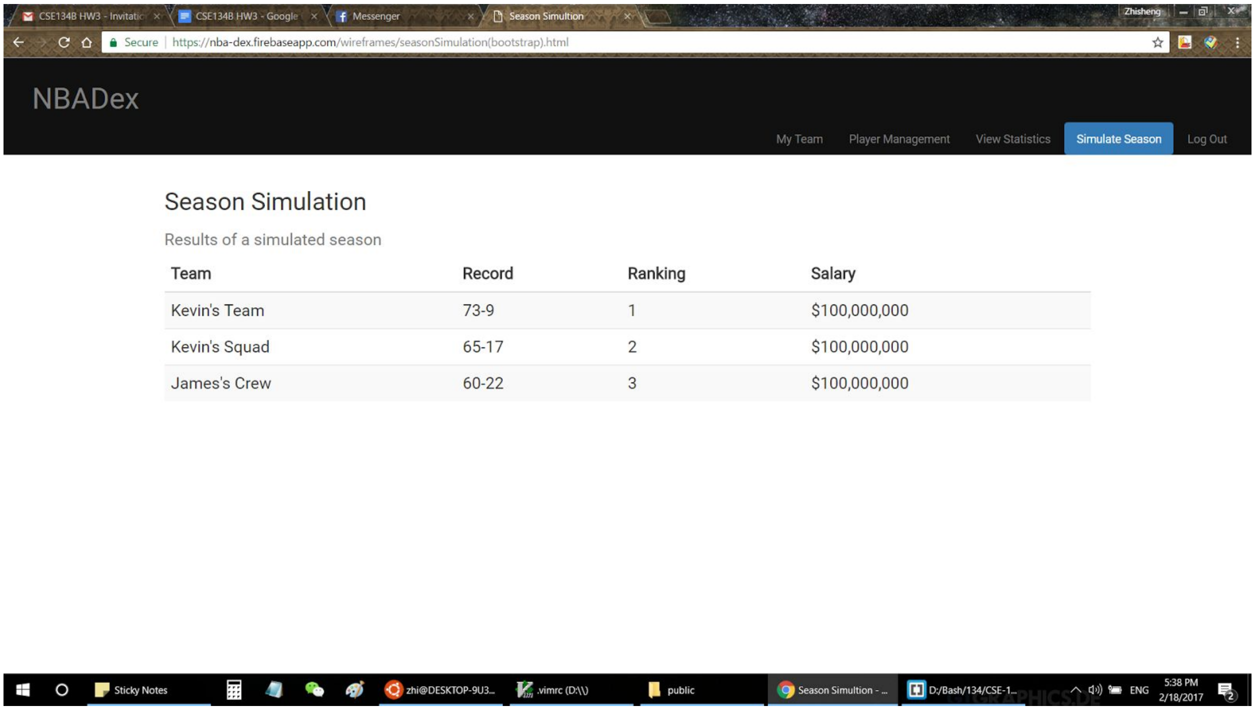
© NBADex

4. Add New Player



Add New Player

5. Simulation Page

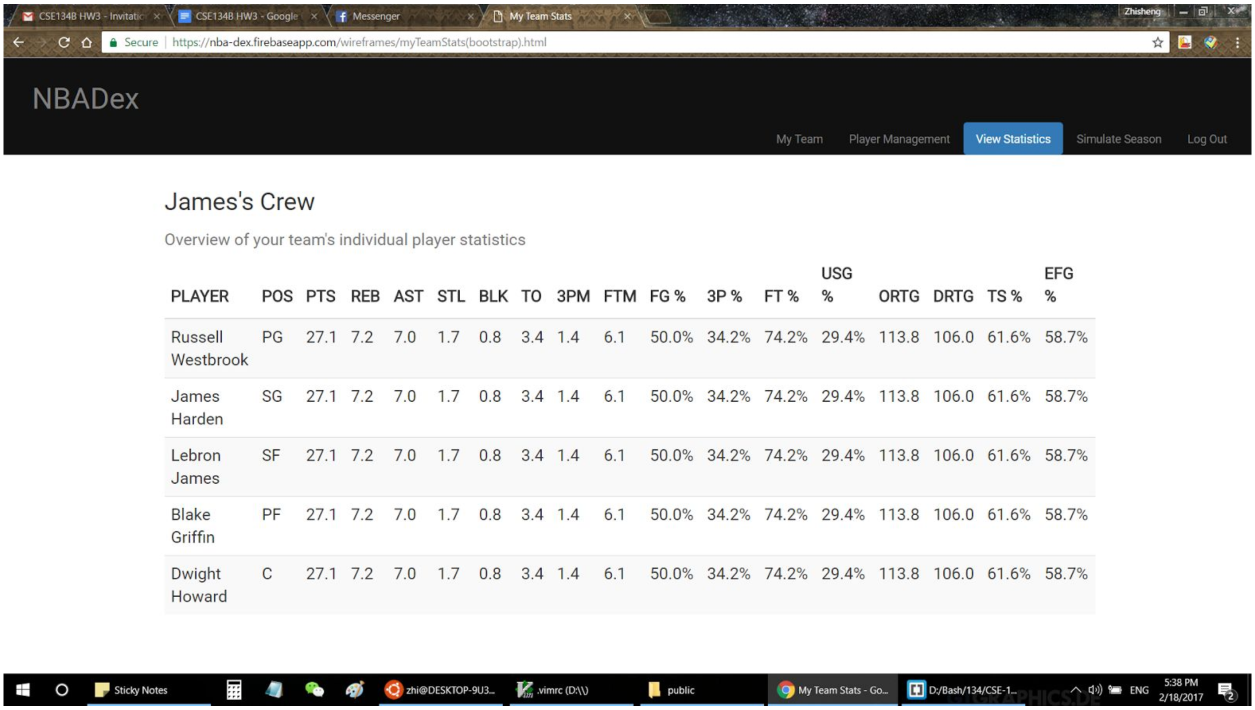


Season Simulation

Results of a simulated season

Team	Record	Ranking	Salary
Kevin's Team	73-9	1	$100,000,000
Kevin's Squad	65-17	2	$100,000,000
James's Crew	60-22	3	$100,000,000

6. Statistic page



James's Crew

Overview of your team's individual player statistics

PLAYER	POS	PTS	REB	AST	STL	BLK	TO	3PM	FTM	FG %	3P %	FT %	USG %	ORTG	DRTG	TS %	EFG %
Russell Westbrook	PG	27.1	7.2	7.0	1.7	0.8	3.4	1.4	6.1	50.0%	34.2%	74.2%	29.4%	113.8	106.0	61.6%	58.7%
James Harden	SG	27.1	7.2	7.0	1.7	0.8	3.4	1.4	6.1	50.0%	34.2%	74.2%	29.4%	113.8	106.0	61.6%	58.7%
Lebron James	SF	27.1	7.2	7.0	1.7	0.8	3.4	1.4	6.1	50.0%	34.2%	74.2%	29.4%	113.8	106.0	61.6%	58.7%
Blake Griffin	PF	27.1	7.2	7.0	1.7	0.8	3.4	1.4	6.1	50.0%	34.2%	74.2%	29.4%	113.8	106.0	61.6%	58.7%
Dwight Howard	C	27.1	7.2	7.0	1.7	0.8	3.4	1.4	6.1	50.0%	34.2%	74.2%	29.4%	113.8	106.0	61.6%	58.7%