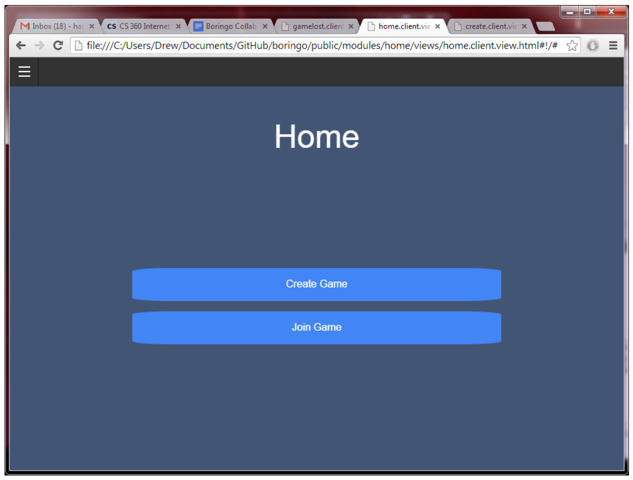
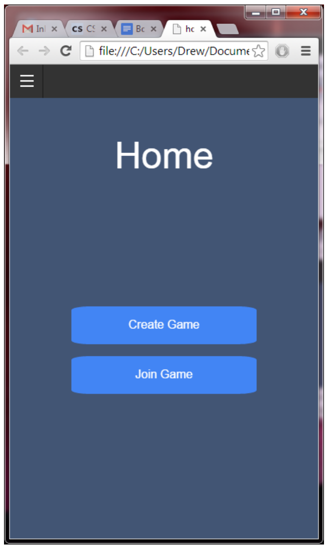
Description:

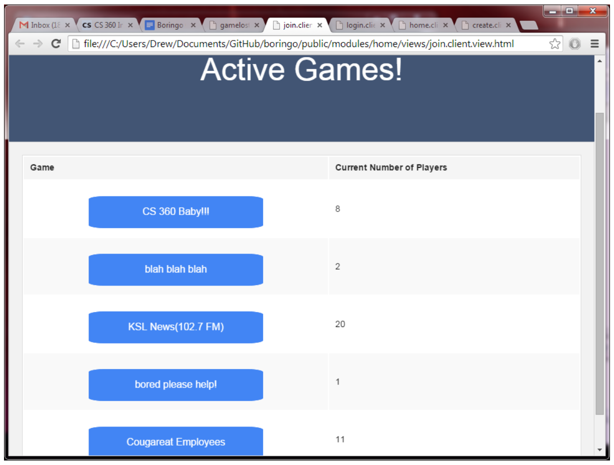
Boringo is intended to be a game designed to bring excitement to those boring meetings or events. Modeled after the game Bingo, it is supposed to contain many customizable features giving users a personalized and fun experience. We worked hard in learning the tools needed to accomplish such a task, however we came to realize that we bit off a lot more than we could chew. We had hoped to have much much more of our features complete, but after spending more than a hundred and fifty total hours on this project we feel pretty accomplished. As of now, we have the server side with the database, controllers, and routers implementing a fully functioning API along with several designs for the pages for which samples are shown in the following pages.


file:///C:/Users/Drew/Documents/GitHub/boringo/public/modules/home/views/home.client.view.html#!/#
Home
Create Game
Join Game



Unfortunately, we weren't able to complete very much of the front-end. This was mostly due to the fact that we were all learning new languages and technologies, and also because we decided to use a boilerplate framework that, while powerful and time-saving in the long-run, turned out to have a very steep learning curve. A handful of major technical problems didn't help either. In spite of these shortcomings, we put in our best effort and have a desire to continue working on Boringo after this semester ends.

Although many of the front end elements could not be implemented to the app, the html was elegantly designed. The pages scaled well, as you can see in the images above.

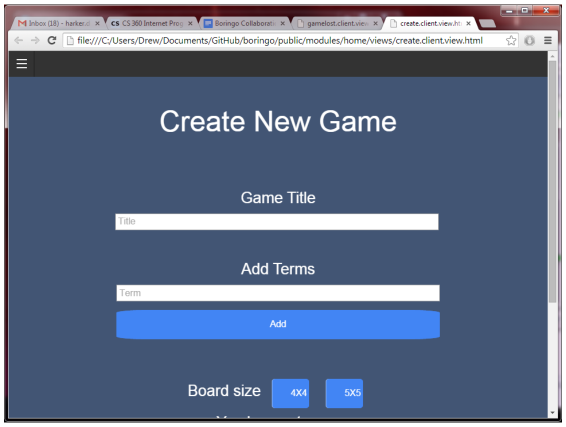


file:///C:/Users/Drew/Documents/GitHub/boringo/public/modules/home/views/join.client.view.html

Active Games!

Game	Current Number of Players
CS 360 Baby!!!	8
blah blah blah	2
KSL News(102.7 FM)	20
bored please help!	1
Cougareat Employees	11

This page is to display the current active local games that a user could join.



file:///C:/Users/Drew/Documents/GitHub/boringo/public/modules/home/views/create.client.view.html

Create New Game

Game Title

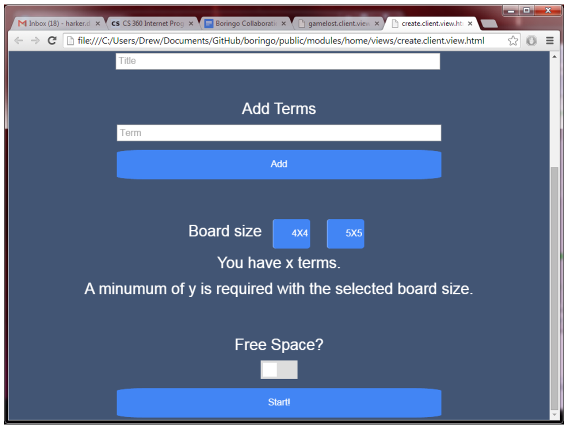
Title

Add Terms

Term

Add

Board size 4X4 5X5



file:///C:/Users/Drew/Documents/GitHub/boringo/public/modules/home/views/create.client.view.html

Title

Add Terms

Term

Add

Board size 4X4 5X5

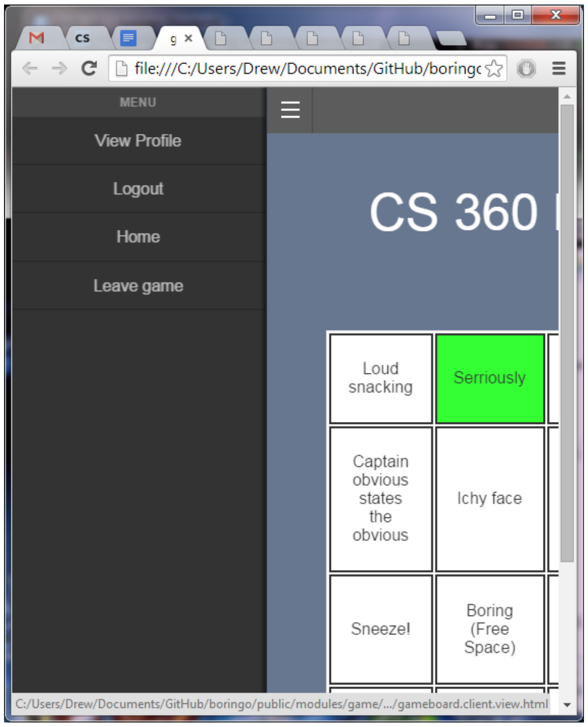
You have x terms.

A minumum of y is required with the selected board size.

Free Space?

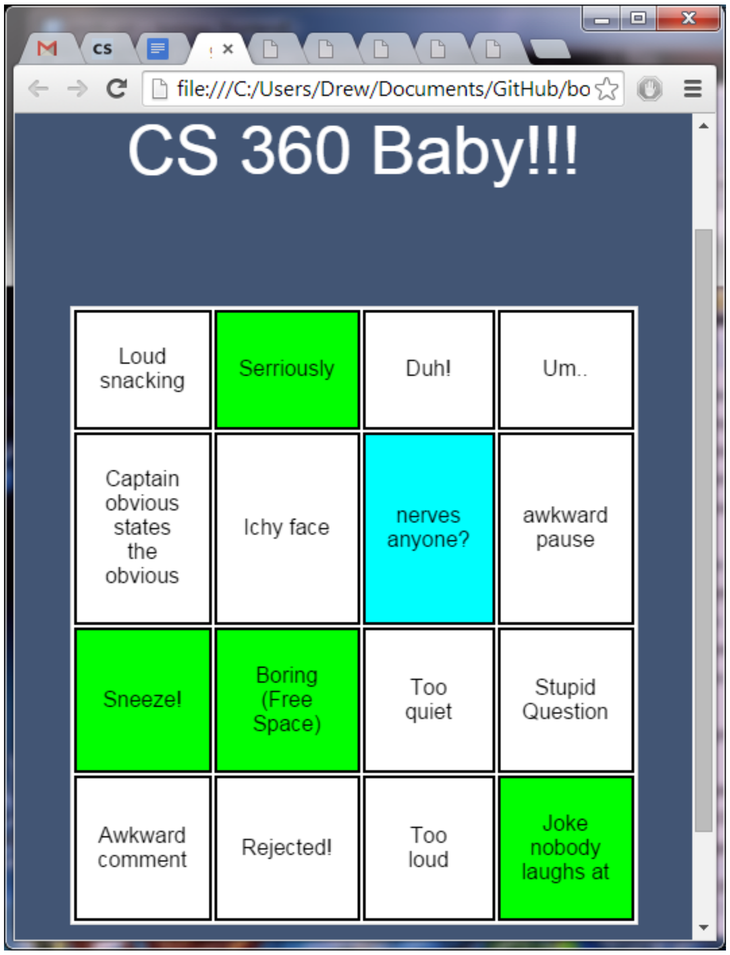
Start!

This page is the create game page, where a user could customize his/her game.


file:///C:/Users/Drew/Documents/GitHub/boringo
MENU
View Profile
Logout
Home
Leave game
CS 360
Loud snacking
Serriously
Captain obvious states the obvious
Ichy face
Sneeze!
Boring (Free Space)
C:/Users/Drew/Documents/GitHub/boringo/public/modules/game/.../gameboard.client.view.html

This screen shot shows the functionality of the off-canvas menus, used to provide easy access to other parts of the app, and, during a game, to show the names of the other players.

The image below is the game board. The green spaces are the ones that the user has already acquired, and blue space is the one his/her cursor is hovering over.



file:///C:/Users/Drew/Documents/GitHub/bo

CS 360 Baby!!!

Loud snacking	Serriously	Duh!	Um..
Captain obvious states the obvious	Ichy face	nerves anyone?	awkward pause
Sneeze!	Boring (Free Space)	Too quiet	Stupid Question
Awkward comment	Rejected!	Too loud	Joke nobody laughs at

Database Schema:

We decided to go with MongoDB because the document database system fit perfectly with how we wanted to organize our project. We broke up our design into User, Board, and Game collections to store the data. We used Mongoose.js to help us manage connections with the database and set up the basic schemas. Below are the schema layouts along with what they contain.

User Collection:

```
var UserSchema = new Schema({
	firstName: {
		type: String,
		trim: true,
		default: '',
		validate: [validateLocalStrategyProperty, 'Please fill in your first name']
	},
	lastName: {
		type: String,
		trim: true,
		default: '',
		validate: [validateLocalStrategyProperty, 'Please fill in your last name']
	},
	displayName: {
		type: String,
		trim: true
	},
	email: {
		type: String,
		trim: true,
		default: '',
		validate: [validateLocalStrategyProperty, 'Please fill in your email'],
		match: [/.+\@.+\..+/, 'Please fill a valid email address']
	},
	username: {
		type: String,
		unique: 'Username already exists',
		required: 'Please fill in a username',
		trim: true
	},
	password: {
		type: String,
		default: '',
		validate: [validateLocalStrategyPassword, 'Password should be longer']
	},
	totalWins: {
		type: Number,
		default: 0,
	},
	totalLosses: {
		type: Number,
		default: 0,
	},
	// The current games that you are playing
	CurrentGames: {
		type: [Schema.ObjectId],
		default: [],
	},
	salt: {
		type: String
	},
	provider: {
		type: String,
		required: 'Provider is required'
	},
	providerData: {},
	additionalProvidersData: {},
	roles: {
		type: [{
			type: String,
			enum: ['user', 'admin']
		}],
		default: ['user']
	},
	updated: {
		type: Date
	},
	created: {
		type: Date,
		default: Date.now
	},
	/* For reset password */
	resetPasswordToken: {
		type: String
	},
	resetPasswordExpires: {
```

```
                type: Date
        }
});
```

Game Schema:

```
var GameSchema = new Schema({
        gameName:{
                type: String,
        },
   // List of terms that can be selected to form a board
   gameTerms:{
        type: [String],
   },
   freeSpace: {
        type: Boolean,
   },
   playerCount: {
        type: Number,
      default: 0,
   },
   players: {
        type: [Schema.ObjectId],
      default: [],
   },
   // In order to keep track of the players in a game and the boards they are connected
to, we store their unique id’s together in this array in pairs.
   boardIdPairs: {
      type: [boardPair],
      default: [],
   },
   boardLength: {
        type: Number,
   },
   winner: {
        type: Number,
      default: -1,
   }
});
```

Board Schema:

```
var BoardSchema = new Schema({
   //An array that hold arrays to keep track of the terms in order for the board
   tiles: [],
   // Keeps track of which tiles have been selected for a given board
   tilesSelected: []
});
```

Future Work:

After we get a fully working basic prototype, we would like to first develop this into a mobile app as we feel that is the way this app will be most used. Once it is on mobile devices, we hope to implement the following options over time:

Game Oriented:

- Leaderboards
- Ability to challenge other players to games
- Public chat / event log
- Crowd validation when selecting tiles and/or overruling victories
- Suggested games based on groups you join, interests/likes on social media, or location (ex. BYU, ACM, Football Game)
- Social media integration (find/invite games with friends)
- Private games
- Real-time progress shown (ex. “John239 is 1 tile away from winning!”)

Lecture and Education Oriented:

- Ability to import and export terms from quizlet
- A “drag and drop” feature for terms to definitions so that during lecture you can connect important definitions