**SecV – Documentation**

**Getting started:**

When starting, in order to set up the initial settings, refer to either of the following websites to install required node dependencies and to create boiler plate code and then replace the files with those in the main directory of the actual game as shown in the next section.

<https://www.danielgynn.com/build-an-authentication-app-using-express-node-passport/>

<https://scotch.io/tutorials/easy-node-authentication-setup-and-local>

Github link to the actual game directory: <https://github.com/sar-ansh/Security-Game>

**Note:** The node modules need to be installed as per the instructions in the websites above. Replacing them will not work. Also apart from node.js, an account needs to be created on [www.mlab.com](http://www.mlab.com) . A username and password along with a url will be generated there which needs to be replaced in main/config/database.js file. The present data in that file works only with a local database.

**Directory tree:**

The directory tree of the main folder is as shown below. Immediately after the structure, each file is described in detail as categorized in the front end and the back end sections.

main

- config

-- auth.js

-- database.js

-- passport.js

- models

-- user.js

- node\_modules

-- bcrypt-nodejs

-- body-parser

-- connect-flash

-- cookie-parser

-- debug

-- ejs

-- express

-- express-session

-- mongoose

-- morgan

-- nodemon

-- passport

-- passport-facebook

-- passport-google-oauth

-- passport-local

-- passport-twitter

-- serve-favicon

- public

-- assets

-- css

--- metro

--- apps

---- chatbox.css

---- explorer.css

---- looknmeet.css

--- nav

---- internet.css

---- tasks.css

---- timeline.css

--- websites

---- games

----- guesswhat.css

----- molebuster.css

----- tictactoe.css

---- bank.css

---- blog.css

---- crackdiy.css

---- gabber.css

---- game\_ads.css

---- mail.css

---- musicfiles.css

---- randomgame.css

--- font-awesome

--- bootstrap.min.css

--- intro.css

--- jquery-ui.css

--- sweetalert.css

-- js

--- apps

---- chatbox.js

---- explorer.js

---- looknmeet.js

--- nav

---- browser.js

---- downloads.js

---- history.js

---- tasks.js

---- timeline.js

--- websites

---- games

----- guesswhat.js

----- molebuster.js

----- tictactoe.js

---- bank.js

---- blog.js

---- gabber.js

---- game\_ads.js

---- mail.js

---- marquee.min.js

---- randomgame.js

---- url\_error.js

--- bootmetro-charms.js

--- bootmetro-panorama.js

--- bootmetro-pivot.js

--- bootstrap.js

--- bootstrap.min.js

--- jquery.js

--- jquery.mousewheel.min.js

--- jquery-ui.js

--- notify.min.js

--- prologue.js

--- prologue.txt

--- start\_menu.js

--- sweetalert.min.js

-- json

--- blog.json

--- chatbox.json

--- downloads\_file.json

--- gabber\_chat1.json

--- gabber\_chat2.json

--- gabber\_chat3.json

--- mail\_inbox.json

--- mail\_sent.json

--- mail\_spam.json

--- mail\_unread.json

--- tasks.json

-- SecV

--- js

---- main.js

---- Boot.js

---- Preload.js

---- MainMenu.js

---- Game.js

--- libs

---- phaser.js

---- phaser.map

---- phaser.min.js

-- favicon.ico

- routes

-- index.js

-- users.js

- views

-- layouts

--- header.ejs

--- footer.ejs

-- apps

--- chatbox.ejs

--- explorer.ejs

--- looknmeet.ejs

-- nav

--- browser.ejs

--- downloads.ejs

--- history.ejs

--- tasks.ejs

--- timeline.ejs

-- websites

--- games

---- guesswhat.ejs

---- molebuster.ejs

---- tictactoe.ejs

--- bank.ejs

--- blog.ejs

--- crackdiy.ejs

--- gabber.ejs

--- gamereviews.ejs

--- mail.ejs

--- musicfiles.ejs

--- randomgame.ejs

--- shop.ejs

--- travel.ejs

--- url\_error.ejs

--- url\_index.ejs

-- about.ejs

-- computer.ejs

-- error.ejs

-- game.ejs

-- index.ejs

-- start\_menu.ejs

- app.js

- package.json

**Note**:

The following things need to be kept in mind when taking the game development further.

1. In order to change the base code and layout of any section, its EJS file needs to be modified in the views folder.
2. In order to change the style of any section its corresponding CSS file needs to be modified in the public/css folder.
3. In order to change the functioning of any section, its corresponding JS file needs to be modified in the public/js folder.
4. In order to change the data that get imported into the game in any section, its corresponding JSON file needs to be modifies in the public/json folder.
5. In order to implement the actual gameplay, the js files need to be modified in the public/SecV/js folder.
6. No other file or folder needs any modification in order to proceed with the game development.
7. New files can always be added accordingly in the appropriate folders in order to keep the code understandable.
8. In order to only implement new tasks, views/nav/tasks.ejs, public/js/nav/tasks.js and public/json/tasks.js are the only files that need to be modified along with occasional changes in the appropriate ejs and js files of the sections for globally defining the variables that will be utilized in that particular task.
9. Instead of including all the js files, css files and json files at the beginning they are defined in the appropriate ejs files to maintain readability and improve the website loading time.

**Game Side (Front end):**

This section will describe all the front end code and modules, their functioning and the workflow.

1. **Views**

On the lines of MVC(Model-View-Controller) framework, this folder comprises of the files that take care of the V part. All the HTML code is present here and the corresponding CSS(Cascading Style Sheets) and JS(Javascript) files are in the public folder as described below. Refer to the public folder.

* 1. **Layouts**

This folder contains the header and footer of the main HTML boilerplate code. In addition, as and where required EJS(Embedded Javascript) is used to embed JS in between the HTML code.

* + 1. **header.ejs**

Apart from the normal HTML code, below mentioned are few notable things in this file.

The bottom nav pane is defined here that has the help and direction button, task unlock button, score board and the hyperlinks to the game, computer and about sections.

Also, Login and Signup mechanism of node is added here. While, Facebook and Google authorization system have been built, they aren’t functional yet as they require oauth linking from developers Facebook/Google account.

* + 1. **footer.ejs**

This file ends the boilerplate code.

* 1. **index.ejs**

This file links the main game, computer and about sections by including the required files as explained below.

* 1. **game.ejs**

This is the game file where gameplay and story mode will take place. It has separate sections for conversations, main menu, prologue and base as explained below.

**Conversations:** This will be the conversation box where in all the dialogues will appear whenever required in the gameplay.

**Main Menu:** This contains the basic options like play/pause, settings, quit game, etc.

**Prologue:** This contains the code which along with the prologue.js file presents the prologue scene of the game. This sets the background of the game and increases the visual appeal.

**Base:** This links to the code in the SecV folder that runs as per the rules of Phaser Game Framework. This sections has the main top down view based gameplay area.

* 1. **computer.ejs**

This file contains the main layout of the in-game computer system. Firstly it defines the left navigation pane that gives the user access to the functionalities such as the browser, downloads folder, history, tasks section, timeline section and others.

Then, on the main screen it includes several files based on the sections using EJS and then manages them using jquery.

* 1. **start\_menu.ejs**

This file defines the layout of the start menu(app section) where the player gets landed when the game is signed in. Here, the type, order, name, color and size of apps can be manipulated. It defined the charms bar and it includes the css file and js files as well that are required for its functioning.

* 1. **about.ejs**

As we lack graphics and a gameplay, playing only the tasks make very less sense. Hence this file contains the disclaimer that the player needs to read for the time being.

* 1. **error.ejs**

This is the file that handles error when a page on the game’s ip address is not found or when there are some internal errors. It can be used to produce custom error messages as well.

* 1. **Nav**

This folder contains the files that define the left navigation pane.

* + 1. **browser.ejs**

This file defines the structure of the in-game browser. Every time a url is entered, through expression matching it redirects to the appropriate page. In order to add new websites, a link to that will need to be provided in this file and it’s JS file as well.

Its CSS and JS is linked in the corresponding CSS and JS files as described in the following sections.

* + 1. **downloads.ejs**

This file defines the structure of the downloads sections. Every time a file is downloaded, it appears in this section where the contents of the file can be seen. All the data shown here is imported from the JSON files.

* + 1. **history.ejs**

This file will define the history section where the player can see the history of websites, apps, etc used by him. It hasn’t been implemented yet.

* + 1. **tasks.ejs**

This file defines the main tasks section from where the tasks need to be initiated. Every time, a new task needs to be added, it needs to be defined here as well in a similar fashion to that of the prior existing tasks.

Its CSS and JS is linked in the corresponding CSS and JS files as described in the following sections.

* + 1. **timeline.ejs**

This file defines the timeline section where the player can visit anytime to see his performance so far. It will show the tasks’ score and will provide a summary of the security concept related to the tasks.

Its CSS and JS is linked in the corresponding CSS and JS files as described in the following sections.

* 1. **apps**

This folder contains the files that define the apps in the start menu(app section).

* + 1. **chatbox.ejs**

This file defines the chatbox section where the player can visit anytime to view his social networking profile and chat with his acquaintances. It needs a lot of upgrades and needs to be modified as per the story.

Its CSS and JS is linked in the corresponding CSS and JS files as described in the following sections.

* + 1. **explorer.ejs**

This file defines the file explorer section where the files in the computer can be viewed. It has all the normal functionalities of an explorer except file handling. It can be improved as and when required.

Its CSS and JS is linked in the corresponding CSS and JS files as described in the following sections.

* + 1. **looknmeet.ejs**

This file defines the dating app section where the player will need to register and then carry on related tasks. It has only the registration page so far.

Its CSS and JS is linked in the corresponding CSS and JS files as described in the following sections.

* 1. **websites**

This folder contains the files that define websites that are the main part of the game.

* + 1. **bank.ejs**

This defines the structure of the website [www.bank.game](http://www.bank.game). The player can visit this website to view the account details of all the characters.

Its CSS and JS is linked in the corresponding CSS and JS files as described in the following sections.

* + 1. **blog.ejs**

This defines the structure of the website [www.blog.game](http://www.blog.game). The player can visit this website to read and write blog posts.

Its CSS and JS is linked in the corresponding CSS and JS files as described in the following sections.

* + 1. **crackdiy.ejs**

This defines the structure of the website [www.crackdiy.game](http://www.crackdiy.game). This was made specifically for a task that needed the player to identify the malicious link.

Its CSS is linked in the corresponding CSS file as described in the following sections.

* + 1. **gabber.ejs**

This defines the structure of the website [www.gabber.game](http://www.gabber.game). This can be visited by the player when he needs to chat with random strangers(chatbots)

Its CSS and JS is linked in the corresponding CSS and JS files as described in the following sections.

* + 1. **gamereviews.ejs**

This defines the structure of the website [www.gamereviews.game](http://www.gamereviews.game). This hasn’t been built yet.

* + 1. **mail.ejs**

This defines the structure of the website [www.mail.game](http://www.mail.game). This is the personal mailbox of the player. He’ll get timely mails from json files as per the tasks here.

Its CSS and JS is linked in the corresponding CSS and JS files as described in the following sections.

* + 1. **musicfiles.ejs**

This defines the structure of the website [www.musicfiles.game](http://www.musicfiles.game). This is another website that is created specifically for tasks.

Its CSS is linked in the corresponding CSS file as described in the following sections.

* + 1. **randomgame.ejs**

This defines the structure of the website [www.random.game](http://www.random.game). It redirects the player to a random in-game game each time. The games are described in the following games folder.

Its CSS and JS is linked in the corresponding CSS and JS files as described in the following sections.

* + 1. **shop.ejs**

This defines the structure of the website [www.shop.game](http://www.shop.game). It hasn’t been implemented yet.

* + 1. **travel.ejs**

This defines the structure of the website [www.travel.game](http://www.travel.game). It hasn’t been implemented yet.

* + 1. **url\_error.ejs**

This defines the structure of the websites where the player will get redirected to every time he enters an invalid url. This page will show a random error message as output.

Its JS is linked in the corresponding JS file as described in the following sections.

* + 1. **url\_index.ejs**

This defines the structure of the index page which has the information about all the safe working websites. It can be referred any time, details about website are needed by clicking the button with book on the browser in the right hand side.

* + 1. **Games**

Several in-game games can be built and integrated for hiding security vulnerabilities and for engaging the player. Right now only three games are developed. Playing these games will boost the main score as well.

* + - 1. **guesswhat.ejs**

This defines the structure of the website [www.guesswhat.game](http://www.guesswhat.game). Its a game where the player enters a range of numbers. The computer then guesses a random number in the range and the player needs to guess the number. Based on the number of guesses, points are awarded.

Its CSS and JS is linked in the corresponding CSS and JS files as described in the following sections.

* + - 1. **molebuster.ejs**

This defines the structure of the website [www.molebuster.game](http://www.molebuster.game). Its a game similar to whack a mole of arcade games where a dot will be randomly jumping and the user will need to hit it. Based on the number of right hits, points are awarded.

Its CSS and JS is linked in the corresponding CSS and JS files as described in the following sections.

* + - 1. **tictactoe.ejs**

This defines the structure of the website [www.tictactoe.game](http://www.tictactoe.game). Its a game where well… this doesn’t really need any introduction.

Its CSS and JS is linked in the corresponding CSS and JS files as described in the following sections.

1. **public**

This is the C (Controller) part of the afore mentioned MVC framework. It containes all the images, js, css, libraries and json files needed for the game.

* 1. **favicon.ico**

This is the favicon file that can always be replaced.

* 1. **assets**

This folder contains all the images that will be needed in the game and the in-game computer system.

* 1. **font-awesome**

This is a library that contains the icons used in the game.

* 1. **metro**

This folder contains the CSS code required for implementing the start menu and its icons. In addition to this folder, start\_menu.css is required as mentioned below.

* 1. **css**

This folder contains all the CSS files required for beautifying the game.

* + 1. **bootstrap.min.css**

This file contains the CSS code required for implementing bootstrap.

* + 1. **jquery-ui.css**

This file contains the CSS code required for implementing jquery.

* + 1. **sweetalert.css**

This file contains the CSS code required for implementing alert system.

* + 1. **intro.css**

This file contains the extra CSS code that beautifies the whole website and all the sections that could not be categorized as below.

* + 1. **start\_menu.css**

This file contains the CSS code required for the start menu (app section).

* + 1. **nav**

This folder has the CSS files required for the sections in the left navigation pane.

* + - 1. **internet.css**

This file contains the CSS code required for browser.ejs, downloads.ejs and history.ejs.

* + - 1. **tasks.css**

This file contains the CSS code required for corresponding tasks.ejs.

* + - 1. **timeline.css**

This file contains the CSS code required for corresponding timeline.ejs.

* + 1. **apps**

This folder has the CSS files required for the apps in the start menu(app section).

* + - 1. **chatbox.css**

This file contains the CSS code required for corresponding chatbox.ejs.

* + - 1. **explorer.css**

This file contains the CSS code required for corresponding explorer.ejs.

* + - 1. **looknmeet.css**

This file contains the CSS code required for corresponding looknmeet.ejs.

* + 1. **websites**

This folder has the CSS files required for the websites.

* + - 1. **bank.css**

This file contains the CSS code required for corresponding bank.ejs

* + - 1. **blog.css**

This file contains the CSS code required for corresponding blog.ejs.

* + - 1. **crackdiy.css**

This file contains the CSS code required for corresponding crackdiy.ejs.

* + - 1. **gabber.css**

This file contains the CSS code required for corresponding gabber.ejs.

* + - 1. **game\_ads.css**

This file contains the CSS code required for corresponding game\_ads.js. (It has no corresponding EJS file.)

* + - 1. **mail.css**

This file contains the CSS code required for corresponding mail.ejs.

* + - 1. **musicfiles.css**

This file contains the CSS code required for corresponding musicfiles.ejs.

* + - 1. **randomgame.css**

This file contains the CSS code required for corresponding randomgame.ejs.

* + - 1. **games**

This folder contains the CSS code required for the browser based in-game games.

* + - * 1. **guesswhat.css**

This file contains the CSS code required for corresponding guesswhat.ejs.

* + - * 1. **molebuster.css**

This file contains the CSS code required for corresponding molebuster.ejs.

* + - * 1. **tictactoe.css**

This file contains the CSS code required for corresponding tictactoe.ejs.

* 1. **js**

This is the most important folder where all the functionalities of the game take place. All the JS files are kept here.

* + 1. **bootstrap.js**

This file contains the JS code required for implementing bootstrap.

* + 1. **jquery.js**

This file contains the JS code required for implementing jquery.

* + 1. **jquery-ui.js**

This file contains the JS code required for implementing jquery-ui.

* + 1. **jquery.mousewheel.min.js**

This file contains the JS code required for implementing jquery’s mousewheel system as required to implement mousewheel scroll in the start menu (app section).

* + 1. **bootmetro-charms.js**

This file contains the JS code required for implementing start menu’s(app section’s) charms bar.

* + 1. **bootmetro-panorama.js**

This file contains the JS code required for implementing start menu’s(app section’s) background scrolling effect.

* + 1. **notify.min.js**

This file contains the JS code required for implementing notification system.

* + 1. **sweetalert.min.js**

This file contains the JS code required for implementing alert system.

* + 1. **start\_menu.js**

This file contains the JS code required for implementing the functionalities of the start menu (app section).

* + 1. **prologue.txt**

This file contains the data that that is fed into prologue.js as explained below.

* + 1. **prologue.js**

This file contains the JS code required for running the prologue scene. It basically takes the text from prologue .txt file and outputs it character by character thus giving the illusion of a bunch of data automatically being typed onto the screen of the in-game laptop.

* + 1. **nav**
       1. **browser.js**

This file contains the JS code required for implementing the functionalities of the browser and in order to add new websites, it needs to be added here as well.

* + - 1. **downloads.js**

This file contains the JS code required for implementing the functionalities of the downloads section.

* + - 1. **history.js**

This file contains the JS code required for implementing the functionalities of the history section. It hasn’t been implemented yet.

* + - 1. **tasks.js**

This is the heart of the game. It’s described in detail at the end of this documentation.

* + - 1. **timeline.js**

This file contains the JS code required for implementing the functionalities of the timeline.ejs. It hasn’t been implemented yet.

* + 1. **apps**
       1. **chatbox.js**

This file contains the JS code required for implementing the functionalities of chatbox.ejs.

* + - 1. **explorer.js**

This file contains the JS code required for implementing the functionalities of explorer.ejs.

* + - 1. **looknmeet.js**

This file contains the JS code required for implementing the functionalities of the looknmeet.ejs.

* + 1. **websites**
       1. **bank.js**

This file contains the JS code required for implementing the functionalities of the bank.ejs.

* + - 1. **blog.js**

This file contains the JS code required for implementing the functionalities of the blog.ejs.

* + - 1. **gabber.js**

This file contains the JS code required for implementing the functionalities of the gabber.ejs.

* + - 1. **game\_ads.js**

This file has several pre-defined advertisements which have HTML and CSS embedded into it and they are disguised as functions. So, new advertisements can always be added here in a similar fashion. And in order to add them to a website, only calling the appropriate advertisement based function with the right parameters below it will suffice.

* + - 1. **mail.js**

This file contains the JS code required for implementing the functionalities of the mail.ejs.

* + - 1. **marquee.min.js**

This file contains the JS code required for adding movements in the advertisements.

* + - 1. **randomgame.js**

This file contains the JS code required for implementing the functionalities of the randomgame.ejs.

* + - 1. **url\_error.js**

This file contains the JS code required for implementing the functionalities of url\_error.ejs.

* + - 1. **games**

This folder contains the JS files required for the in-game games.

* + - * 1. **guesswhat.js**

This file contains the JS code required for implementing the functionalities of the guesswhat game.

* + - * 1. **molebuster.js**

This file contains the JS code required for implementing the functionalities of the molebuster game.

* + - * 1. **tictactoe.js**

This file contains the JS code required for implementing the functionalities of the tictactoe game.

* 1. **json**

This folder contains all the json files required for dynamic flow of the game. Instead of hardcoding the data and embedding it into the game, it is stored in json files as described below in order to make updating the game easier.

**Note:** Everytime the user enters some data, it also needs to be added in the appropriate json file at the right place. But this functionality hasn’t been added yet.

* + 1. **blog.json**

This file contains the posts that appear on the [www.blog.game](http://www.blog.game) website.

* + 1. **chatbox.json**

This file contains the replies the the chatbot outputs based on the user input using. This works using regular expressions.

* + 1. **downloads\_file.json**

This file contains the data that is downloaded during the tasks. At time, its contents may help the player to proceed in the game.

* + 1. **gabber\_chat1.json**

This file contains the replies that the player gets when chatting with strangers on [www.gabber.game](http://www.gabber.game). This file also works based on regular expressions. Also in order to create randomness, three such files are created and each time the user is connected to a random chatbot based on these files.

* + 1. **gabber\_chat2.json**

This files works just like gabber\_chat1.json.

* + 1. **gabber\_chat3.json**

This file works just like gabber\_chat3.json.

* + 1. **mail\_inbox.json**

This file contains the messages that need to be shown in the inbox section of [www.mail.game](http://www.mail.game).

* + 1. **mail\_sent.json**

This file contains the messages that need to be shown in the inbox section of [www.mail.game](http://www.mail.game).

* + 1. **mail\_spam.json**

This file contains the messages that need to be shown in the inbox section of [www.mail.game](http://www.mail.game).

* + 1. **mail\_unread.json**

This file contains the messages that need to be shown in the inbox section of [www.mail.game](http://www.mail.game).

* + 1. **tasks.json**

This file contains the data related to the instructions and the directions of the tasks that will be shown to the player every time a new task is initiated.

* 1. **SecV**

This is the main gameplay folder that contains all the files that will run the main game. This folder corresponds to Phaser Game Framework. Its very easy to understand and to implement in order to create a HTML5 based game.

In order to understand the working of this directory, 2-3 examples/tutorials of Phaser based small games are advised. They can be easily found on the internet.

* + 1. **js**
       1. **main.js**

This is the main Phaser file that initiates the gameplay and links it to the base section in the game.ejs file. Also it links the other states of the game as described below.

* + - 1. **Boot.js**

This file defines the screen size, alignment, number of pointers, etc.

* + - 1. **Preload.js**

This file loads all the assets and names them.

* + - 1. **MainMenu.js**

This file defines the main menu functionality of the Phaser based game div.

* + - 1. **Game.js**

This is the main file where all the map, characters, positions and their collision functions will need to get defined.

* + 1. **libs**

This folder contains the libraries required for running the Phaser Game Framework.

**Server Side (Back end):**

Every other file in the main directory is a node.js related file. In order to understand their workflow, refer to the two links mentioned at the top of this documentation.

**tasks.js:**

All the tasks are defined here.

Consider for example 5th task of to Chapter2. In order to make it work, a function called task2\_5 is created which changes the values of the global variables defined in the appropriate section’s js file(like mail.js or gabber.js). This change of value initiates the task and makes the help and direction button appear. Now, in those js files, after the task completing link/button are clicked, a function called check2\_5 is called by passing the newly defined variables that checks whether or not the task is completed and then increments score accordingly and hides the helping buttons. Also this unlocks the next task.

All the task functions work in a similar fashion.

The data for help and direction buttons is getting imported from appropriate json file.

scoreAnim presents the addition of score in an animated fashion.

unlockTask function unlocks the next task.

hideHelp hides the help buttons.

In order to add new tasks, apart from creating these functions, in the tasks.ejs file, another block/div needs to be created for the new task just like those above it.