SOFT252 Report

1 Functionality
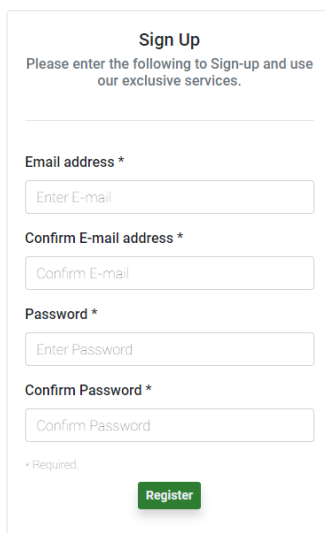
1.1 What does the application do?

Overview: To assist in creating a developer presence on the internet, in which the admin (developer) can upload/edit/remove portfolio and blog pieces via the dashboard. Communication between the client and the developer with an instant messaging service. Registration and login services, managed by the admin in the dashboard, that includes removing and banning accounts.

1.2 How do users interact with it?

1.2.1 User Accounts

Registration

User must first click the Sign-Up button located in the top right corner and then fill out the fields. E-mail must be an E-mail and not in use already on the website. Password must be more than 8 characters.

Login

Firstly, have a registered account and then press the Sign In button located next to the Sign-up button. To login as admin use liamwest1@hotmail.com as the E-mail and then qwerty123 as the password, this will give the user access to the dashboard.

1.2.2 Manage Blogs/Portfolio

Create/Edit/Remove Blog and Portfolio

Firstly Sign-In and navigate to the dashboard in the navbar. Click Manage Blog and fill in the fields required. (All fields) and then click Submit. Similarly, for editing, however in the other tab it is possible to select a post by title the user wants to edit. By clicking remove, the blog will be deleted from existence. This works the same for managing portfolio pieces.

1.2.4 Manage Users

Ban/Remove Users

It is important to update the list of users if the admin has been idle on the page for too long. This is necessary if the admin wants to view recently registered accounts. Remove user button removes the account from existence while the Ban user button makes it, so they cannot Sign up again with that E-mail.

Users

lol@gmail.com
thisatest@gmail.com
liamwest1@hotmail.com

Update List

Remove User

Ban User via E-mail
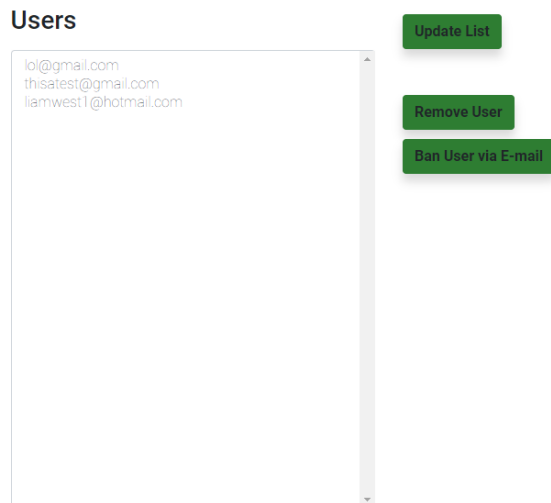
1.2.5 Instant Messaging

Admin/Normal User

Clicking on Live chat now while Signed in to admin will display a different view to the client. Showing a list of users with the messages sent to a from. On the other hand, the normal user is just shown the chat log.

Chat

Select Chat

Send

Live Chat Now

Chat

[Mon Dec 31 2018 12:13:58 GMT+0000 (Greenwich Mean Time)]
liamwest1@hotmail.com: This is a message to lol@gmail.com

[Mon Dec 31 2018 12:13:58 GMT+0000 (Greenwich Mean Time)]
liamwest1@hotmail.com: This is a message to lol@gmail.com

[Mon Dec 31 2018 17:45:11 GMT+0000 (Greenwich Mean Time)]
lol@gmail.com: This is a test

Send

Live Chat Now
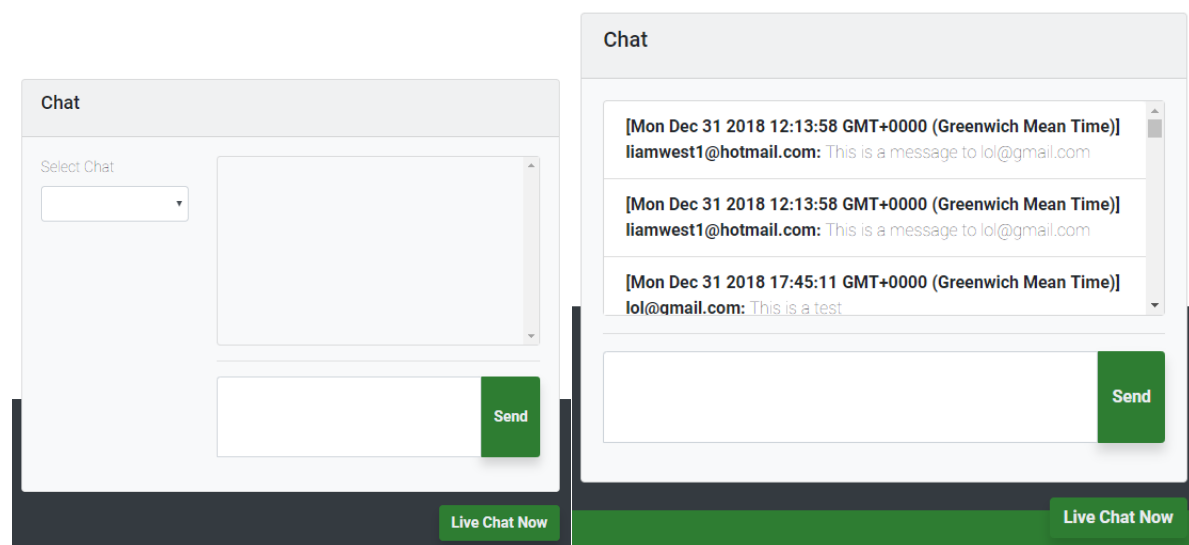
1.3 What technologies were included?

- NodeJS
    - Bcryptjs
        - Hashes and salt the user password.
    - Body-parser
        - Used with express to get data from the body of the Document Object Model.
    - Cookie-parser
    - Cucumber
        - Tests Use Cases against the system.
    - Debug
    - Express
        - Handles server side, get and post requests.
    - Express-handlebars
        - The view. Ability to input data from the server into the view without refresh.
    - Express-socket.io-session
        - Used to parse express-session data to socket.io.
    - Express-validator
        - Validates information such as Password length, E-mail verification and adds errors to an array.
    - http-errors
    - Mongodb
        - Database to store user data, messages, blogs and portfolio items.
    - Mongoose
        - MongoDB helper.
    - Morgan
    - Multer
        - Used to store files, accessible in the public folder. The path will be stored in the database for access from the view.
    - Serve-favicon
    - Socket.io
        - Used for user to user communication. Between the admin and users. A many to one relationship.
    - Express-generator
        - Generated express folder structure.
- Data Storage
    - MongoDB
    - Mongoose
    - MongoDB Atlas
        - Used to store collections and documents online.
- Other JavaScript plugins
    - Dotdotdot.js
        - Applied an ellipsis to content after assigned amount of lines. This was implemented into the thumbnails for blogs/portfolios.
- HTML
    - HTML5 Boilerplate

- Was originally used as the folder structure for the website before switching to express. Some parts of the folder files are still included.
    - o FontAwesome
        - A selected of icons with varying sizes.
    - o GoogleFonts
        - These fonts are applicable for a good user experience.
    - o ARIA
        - The content of this website will be readily available for those with disabilities, making it accessible for all.
- CSS
    - o Bootstrap 4
        - Responsive helper classes so that the website can be designed for both mobile and desktop.
- JS
    - o jQuery
        - A more efficient form of Javascript.

2 Requirements

2.1 Who was the app aimed at?

The app is aimed specifically for a developer who requires an online presence, a website to share content and update potential clients on work. It is also used as a viewing and messaging service for others considered as potential freelance clientele for the developer. See personas for a more detailed analysis of users.

(See 8.1)

2.2 What features were included and why?

The app needed to include a dashboard so that the admin could create/edit/remove blogs and portfolio items to consistently update the potential clients and interested viewers. The app also needed a Registration and Login service so that 3rd party services could not access the instant messaging service. Rather the users that have a verified E-mail will be granted access. Also, in future adaptions the comment section will be for the users. These users needed to be managed in the dashboard with the Manage Users feature. See Use Cases for a more detailed analysis of features.

Folder destination:

SOFT352 > Node > Cucumber > features > {Feature_Name}.feature (file extension)

2.3 Functional, e.g., unit, testing

Test Description	Expected Result	Test (Passed/Failed)
Sign Up Success	User is now able to login with credentials.	Passed.
Sign Up Fail	Error messages displayed after	Passed.

	using a password shorter than 8 characters.	
Add Blog	Blog is displayed on blogs page after being added.	Passed.
Add Portfolio	Portfolio is displayed on portfolio page after being added.	Passed.

3 Design

3.1 What processes run where (clients/servers/peers)?

3.1.1 Server

- **Index.js**
  - Router.get('/')
  - Router.get('/blog)
  - Router.get('/blogpost/:id')
  - Router.get('/portfolio')
  - Router.get('/portfolioitem/:id')
  - Router.post('/signout')
  - Router.get('/signin')
  - Router.post('/signin')
  - Router.get('/signup')
  - Router.post('/signup')
  - Router.get('/admin')
  - Router.post('/admin/getusers')
  - Router.post('/admin/removeusers')
  - Router.post('/admin/banusers')
  - Router.post('/admin/addblog')
  - Router.post('/admin/getblog')
  - Router.post('/admin/editblog')
  - Router.post('/admin/deleteblog')
  - Router.post('/admin/additem')
  - Router.post('/admin/getitem')
  - Router.post('/admin/edititem')
  - Router.post('/admin/deleteitem')
  - Sockets.init
    - Socket.on('connection')
    - Socket.on('new user')
    - Socket.on('input')
    - Socket.on('display')
    - Socket.on('disconnect')
    - (This is used to handle emit from client for messaging and sends emit to respective clients.)

- **App.js**
  - Applies all middleware.
- **www.js**
  - Server listens on port.

3.1.2 Client

- Admin.js
  - Clears fields on manage blog/portfolio.
  - Populates blog list (edit post)
  - Populates portfolio list (edit item)
  - Populates user list
  - Ajax calls to ban selected user.
  - Ajax calls to remove selected user.
  - Ajax calls to remove blog post.
  - Ajax calls to remove portfolio item.
- adminDatePicker.js
  - Provides 3rd party library code for the date pickers on the admin page.
- Client.js
  - Sends input message to server.
  - Admin can select user from a list. Displays depending on the selected user.
  - Loads messages.
  - Updates user list.
  - Updates scroll to start at the bottom.
- Comments.js
  - Not implemented fully.
  - Displays counter.
  - Submit button display transition.
- Ellipsis.js
  - 3rd party plugin for thumbnail content line cap.

3.2 How do the processes interact?

The client sends data to the server.

The server either sends data back to the client or applies data to routes and redirects the client.

3.3 How are the data and code structured?
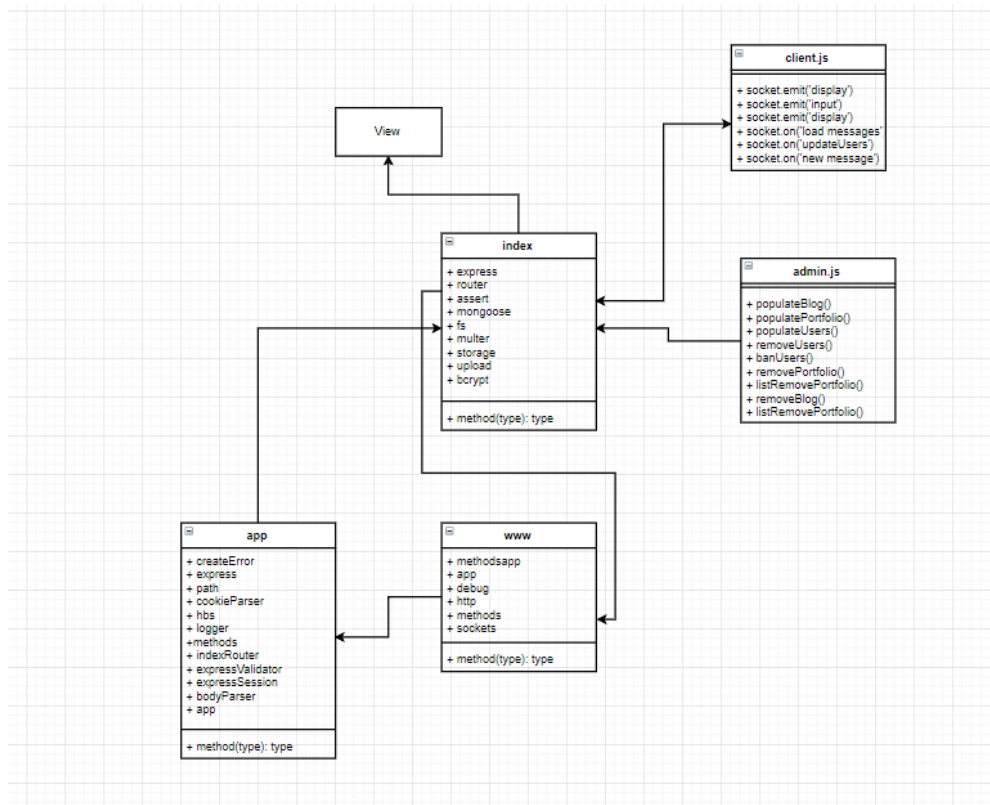
3.4 Why are these structures appropriate?

The view is separated from the model through a controller (MVC) using the MEAN stack. The server takes care of gathering and sending data with MongoDB. Get and Post requests from the client are handled by express and either sends data back to the client or updates the render of the session with additional data, such as when Signing in. Instead of Angular I have chosen to use Handlebars.

This structure is important as the front-end will be separated from the back-end and therefore becomes a more modular system.

4 Performance

4.1 How effective were the user interaction mechanisms (User Testing)
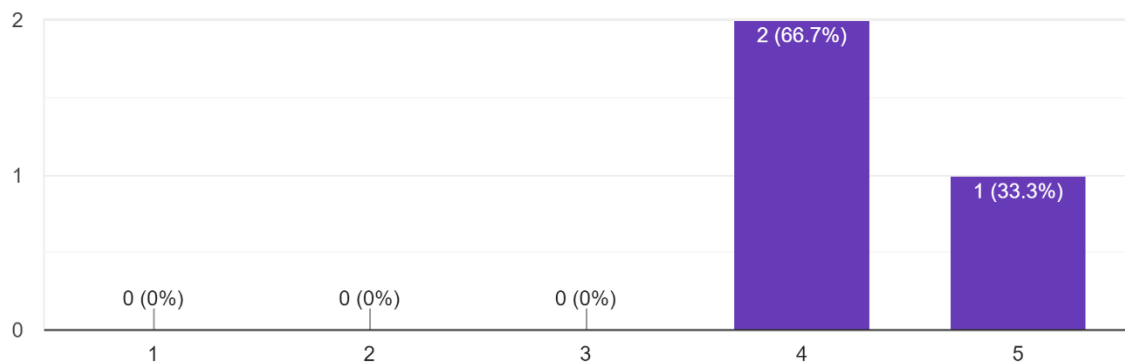
The users interacted with ease, however certain parts of the system can be improved. I chose a few candidates based on user personas to answer a questionnaire. (See 8.4 User Tests)

Navigation

The navigation was satisfactory with the navigation being clear.

Rate how easy it was to navigate.
3 responses

Instant Messaging

- Unclear usernames.
- Dates are a bit too long to read.

Managing Blogs/Portfolio

- Previewing blogs required.
- User feedback after post submitted.
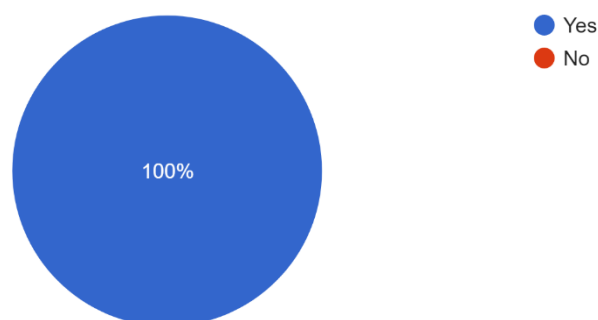
Manage Users

- User list needs to update dynamically.

Validation

Validation errors are clear to the users.

Do you understand validation errors when entering incorrect data?
3 responses

Registration

- Needs confirmation E-mail.

4.2 How responsive is the application?

The application response time did not have any critique from the usability testing, and therefore I can presume the performance is high.

4.3 How quickly does it load?

Index:

36 requests | 11.6 KB transferred | Finish: 608 ms | DOMContentLoaded: 336 ms | Load: 501 ms

Blog:

33 requests | 11.2 KB transferred | Finish: 536 ms | DOMContentLoaded: 265 ms | Load: 509 ms

Portfolio:

36 requests | 11.6 KB transferred | Finish: 487 ms | DOMContentLoaded: 264 ms | Load: 380 ms

Sign-in:

31 requests | 19.1 KB transferred | Finish: 462 ms | DOMContentLoaded: 262 ms | Load: 366 ms

Sign-up:

32 requests | 11.3 KB transferred | Finish: 495 ms | DOMContentLoaded: 295 ms | Load: 376 ms

Dashboard:

38 requests | 14.5 KB transferred | Finish: 534 ms | DOMContentLoaded: 359 ms | Load: 456 ms

4.4 How heavy/light is the network load generated?

Network load is reasonably light. Get users request is unusually high and could be adjusted for better performance.

Name (page)	Request Type	Load time (ms)
Index	Get	98.010
Blog	Get	72.576
Portfolio	Get	115.670
Signin	Get	7.014
Signup	Get	9.339
Signin	Post	133.951 (success) or 205.270 (fail)
Admin	Get	107.235
GetItem	Post	34.594
GetPost	Post	39.583
GetUsers	Post	273.810

4.5 How robust is your application to failure and unreliable users?

A problem that occurred late into development was that if the user does not logout at the end of their session then they will not be able to log back in. This is due to MongoDB saving their status and not changing it after their session is destroyed or the server crashes.

5 Development Process

5.1 Description of the development process.

A ScrumBan approach was used for development using various DevOps techniques. ScrumBan is a mix between Scrum and Kanban methodologies, usually utilized in teams, however this iterative approach suited the project well, with consistent functionality testing and changes to the backlog. This was more suitable than the waterfall methodology. This was utilized on Trello, moving items from the backlog, to planning, splitting items into releases, and sprints to be complete within a 2-week period at a time. This gave clear, concise milestones to push for.

5.2 Description of the tools and technologies supporting the development process.

5.2.1 Project Management

- Trello
    - Link: https://trello.com/b/ebuqrUmq
    - This tool manages the backlog of items and follow a ScrumBan approach with moveable cards.
- Elegantt for Trello
    - A plugin used with Trello to create a Product Roadmap. A roadmap provides an overview for release/sprint deadlines.
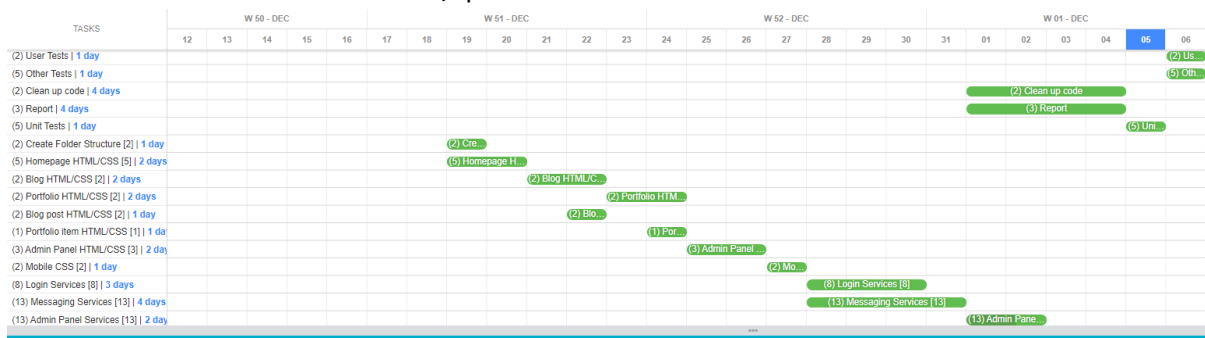
- Scrum for Trello
    - A plugin used with Trello to support a ScrumBan flow, assigning values to cards for predicted hours, and hours complete.
- Burndown Chart for Trello
    - Link: https://BurndownForTrello.com/share/fyxrvvd9ij
    - A plugin that goes together with Scrum for Trello, creating a burndown chart depending on the values from the hours complete. This predicts the date of completion with a graph demonstrating where the project stands compared to where it should be.
- GitHub
    - Link: https://github.com/lwest2/SOFT352

- o GitHub provides a repository service and version control. Commits and pushes allow the data to be pulled from other systems. Rollback operations can be used to undo previous commits therefore data cannot be lost.
- GitHub for Desktop
  - o A Desktop client for GitHub repositories with a user-friendly interface for push and pull requests.
- DiffMerge
  - o Used in conjunction with the merge of branches. If there are any conflicts that cannot be resolved by GitHub then changes can be edited with DiffMerge.
- Atom
  - o Used to write the source code for HTML, CSS and JavaScript/JQuery.

5.2.2 Documentation

- Sketchpad 5.1
  - o A creation tool for drawing the user persona images.
- Balsamiq Mockups 3
  - o A tool to create sketches/prototypes of the webpage design. Also used to represent a tree hierarchy of pages.
- Microsoft Word & PDF
  - o Used for documenting reports/test results/personas/user stories.

5.2.3 DevOps

- Jenkins
  - o Used as a service to test build against Use Tests (Cucumber) and Unit Tests (Q
  - o Unit) and build depending on success.
- Ngrok
  - o Used to make local IP accessible to public so that Jenkins (Hosted on 127.0.0.1:8080) could read Git Commits using Webhooks.

5.2.4 Testing

- QUnit
  - o Used to implementing assert testing for various parts of the website.
- Cucumber
  - O Used to implement Use Case testing.

6 Personal Reflection

To conclude, the express framework, using the MEAN stack was the most important feature to implement as it gave an organized approach to the back-end, and a way to update the front-end dynamically. Adding blogs and portfolio pieces was more efficient than I had previously anticipated. Previous plans were to have individual pages for different articles however with Handlebars I was able to update a single page dynamically depending on the selected item.
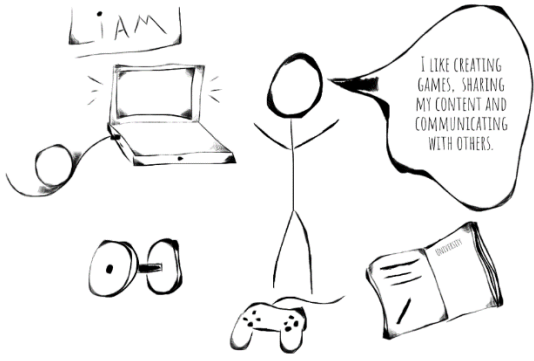
On the other hand, the registration could have been implemented with an E-mail confirmation however I could not do this due to time constraints. This feature would be released if the website deploys to the web. Research suggests that the use of PassportJS would have been the preferred approach to user authentication, and in future projects I will dive into this area.

My Developer Operations could have been improved with CI (Jenkins) properly integrated with my cucumber tests and QUnit tests, I would have then included feature flagging for feature releases to a percentage of users. These are things I would like to improve on in the future.
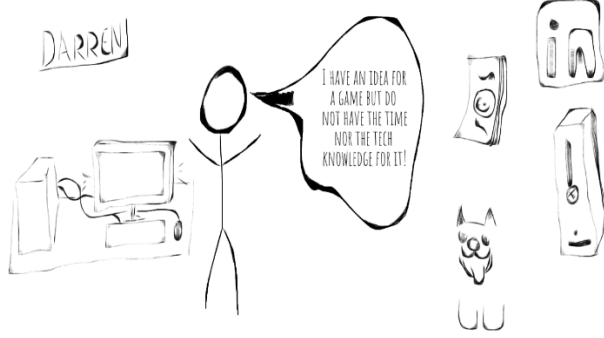
8 Appendices

8.1 Personas

Persona 1 – Liam

	Behaviours - Uploads games that he has created to the web. - Blogs about his development process. - Communicates with clientele. - Advertises himself to find more clientele.
Facts - Age: 21 - Studies Computing with Games Development. - Lives in England. - Is motivated and organised.	**Goals & Needs** - Be a known games developer in the industry. - Become a better developer. - Be organized. - Have a professional developer profile.

Persona 2 – Darren

	Behaviours - Browses the web when bored. - Looks for connections via LinkedIn to develop his game idea. - Games a little but not too much.
Facts - Age: 25 - Works a full-time job for accountancy. - Has a dog.	**Goals & Needs** - Communicate with games developers that will make his game.

- Plays Xbox. - Has a LinkedIn account. - Likes money.	- Review developer blogs and portfolios. - Discover social medias for communication.

Identification of Primary Features based on Personas

[Persona 1 – Liam]

Feature:

Create Portfolio.

Why do they need the feature?

So that the developer can present his work to potential clients.

Other apps that use this feature:

http://www.iamdeantate.com/bioshock/

Details: Each page on this website has an extensive amount of detail for every title created. This is the main attribute of the website.

http://www.pierre-stempin.com/

Details: Linear style portfolio with navigation that scrolls down the page instead of opening a new page. Has all the information and pictures embed into one page.

http://www.yanknoop.com/

Details: Simplistic grid design of pictures representing portfolio pieces. Clicking on these leads the user to separate pages showing details of the project, and media.

Feature:

Create Blogs.

Why do they need the feature?

So that they can organize and advertise their work over the development cycle.

Other apps that use this feature:

https://en.blog.wordpress.com/

Details: WordPress is known for its modular, customizable approach to blog pages. This has all the features that I require for my own blog page, recent dates, comments, blog posts, search queries etcetera.

https://brainybeard.co.uk/blog/

Simple blog layout with linear page layout, with buttons that lead to other pages related to the blog posts.

Feature:

Chat communication.

Why do they need the feature?

Useful for the client and the admin to communicate with eachother through instant messaging services.

Other apps that use this feature:

https://support.microsoft.com/en-gb/contactus/

Details: Microsoft Contact Us app connects you to staff after initializing the conversation. The connection closes after closing the window.

https://www.facebook.com/

Details: Use instant messaging via a friends list. Keeps track of conversations and notifies when there is a new message.

8.2 User Stories

[ADMIN]

As a…

Admin

I need…

Add blog posts

So that…

The site can be consistently updated with new blog posts.

As a…

Admin

I need…

Remove blog posts

So that…

Blog posts can be taken down without going into the code.

As a…

Admin

I need…

Edit blog posts

So that…

Existing blog posts can be updated with information, or in case the wrong information was added.

As a…

Admin

I need…

Remove users

So that…

Spam accounts or accounts that have caused offense can be deleted.

As a…

Admin

I need…

Add portfolio items

So that…

The site can be consistently updated with new portfolio items.

As a…

Admin

I need…

Remove portfolio items

So that…

Portfolio items can be taken down without going into the code.

As a…

Admin

I need…

Edit portfolio items

So that…

Existing portfolio items can be updated with information, or in case the wrong information was added.

As a…

Admin

I need…

Remove blog comments

So that…

Offensive comments or spam can be moderated.

As a…

Admin

I need…

Accept chat requests

So that…

Communication with clients can be queued and spoken to after accepting.

As a…

Admin

I need…

Communicate with client

So that…

Message the client over a short period of time. Chat history should be saved.

As a…

Admin

I need…

Decline chat requests

So that…

Due to a known repeat spammer, or other reasons the admin should be able to decline the chat request.

As a…

Admin

I need…

Reply to comments/Comment

So that…

The admin can comment on own posts or reply to discussion on blog posts.

As a…

Admin

I need…

Sign in

So that…

I can get to the admin panel. Account should be pre-created with username and password.

[USER]

As a…

User

I need…

Comment on blog posts

So that…

I can discuss topics with other users and the administrator/developer.

As a…

User

I need…

Edit/Remove own comments

So that…

Any mistake made can be reverted.

As a…

User

I need…

Navigate the website

So that…

I can get between pages such as blogs, portfolio, and home page.

As a…

User

I need…

Chat with admin

So that…

I can discuss further means of discussion and projects for commission.

8.3 Sketches

Page Hierarchy
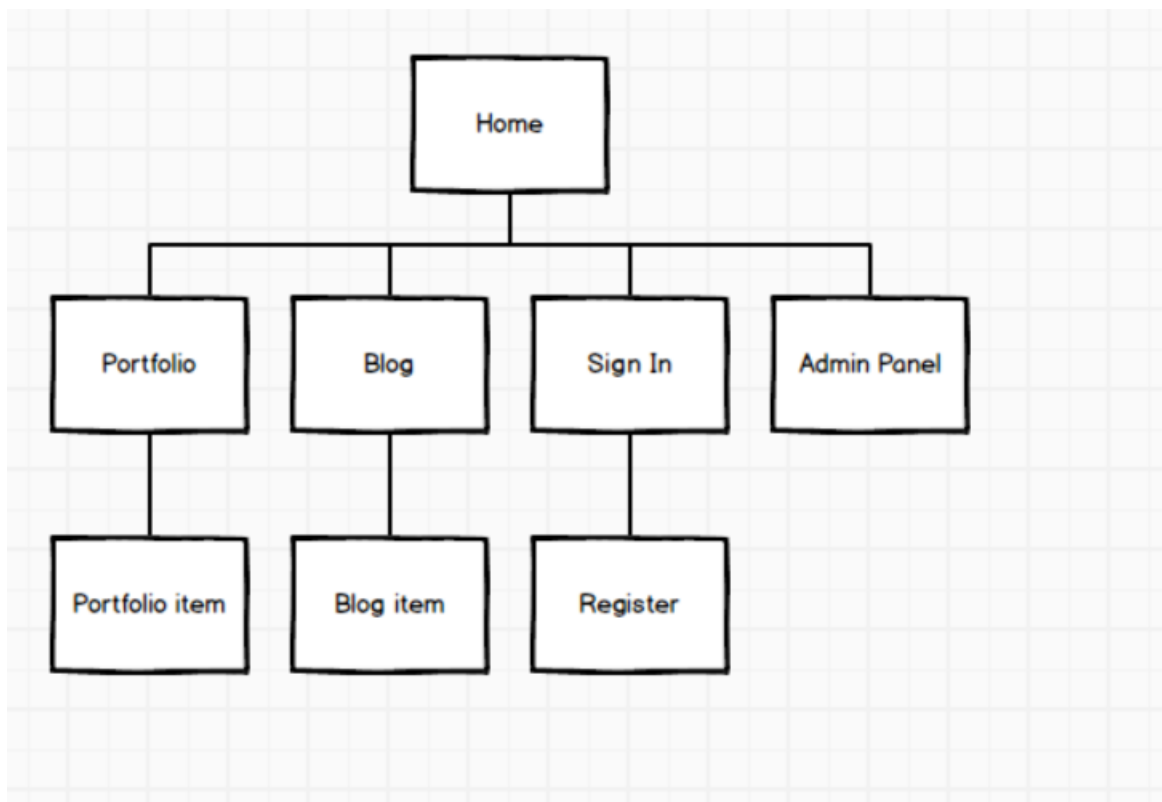
Instant Messaging
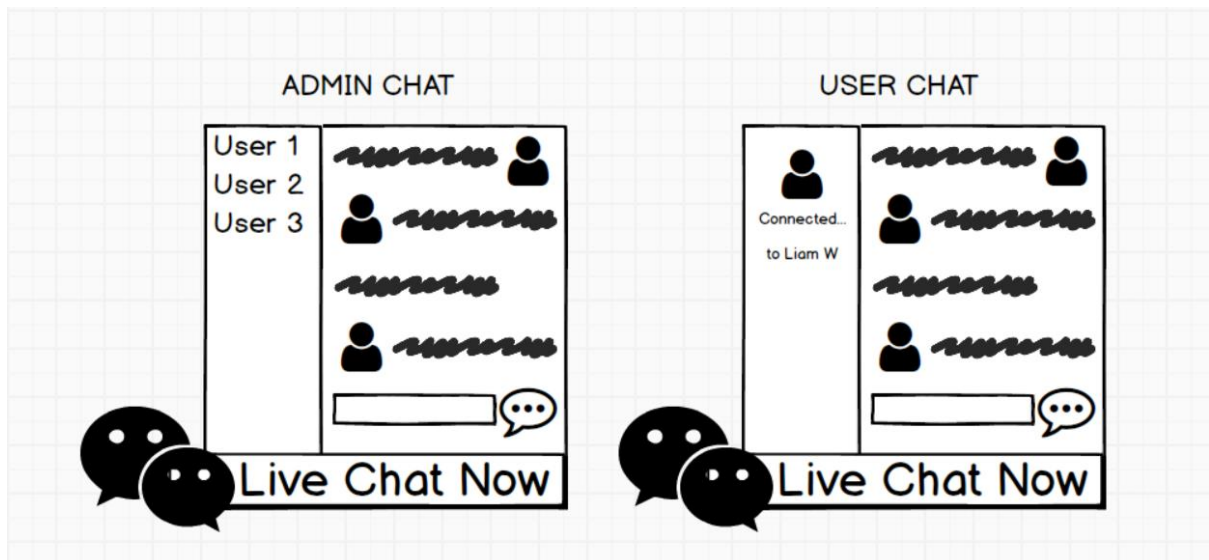
Homepage
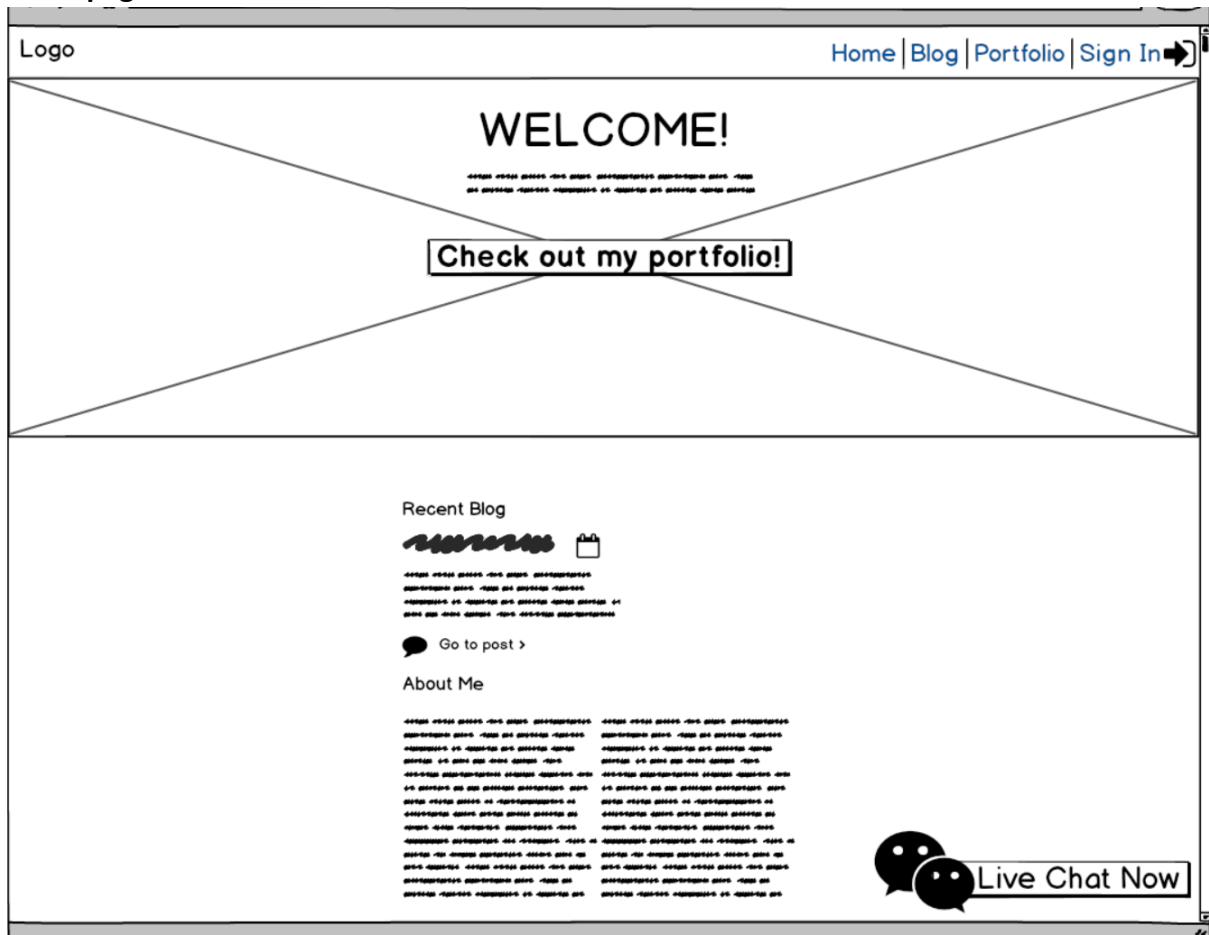
Portfolio page
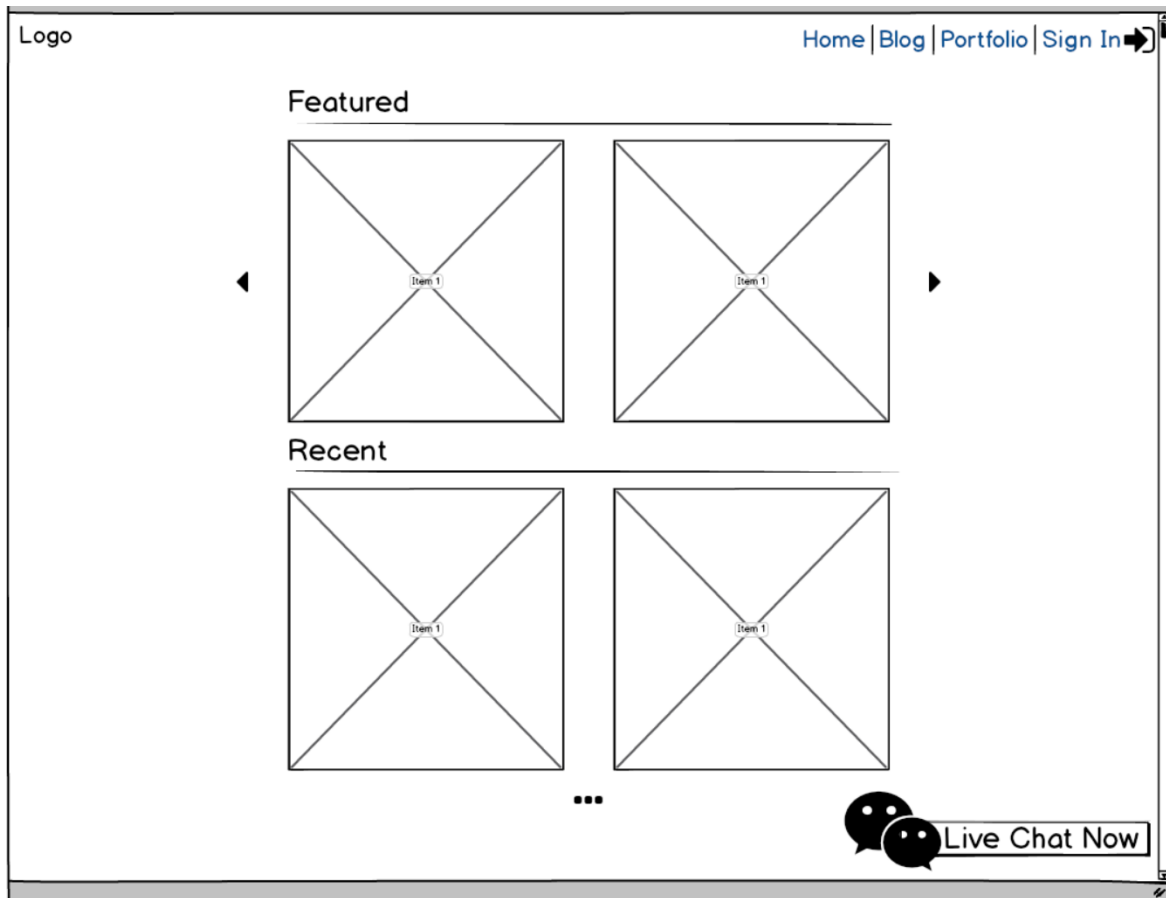
Blog page

Admin – Manage Users

Logo   Home | Blog | Portfolio | Sign Out

Manage blogs

Manage users

Manage portfolio

User 1 Date Joined
User 2 Date Joined
User 3 Date Joined

Remove Selected ›

Chat Requests

Admin – Manage Portfolio – Create Portfolio

Logo

Home | Blog | Portfolio | Sign Out

Manage blogs

Manage users

Manage portfolio

Create portfolio item | Edit existing item

Edit Existing

Blog Post

Get Item Data › Remove Item ›

Title

Title...

Content <HTML>

Text input...

Date

Thumbnail

Submit ›   Clear

Chat Requests

25

Admin – Manage Portfolio – Edit Portfolio

Logo

Home | Blog | Portfolio | Sign Out

Manage blogs

Manage users

Manage portfolio

Create portfolio item   Edit existing item

Create Item

Title

Title..

Content <HTML>

Text input..

Date

/ /

Thumbnail

Submit ›   Clear

Chat Requests

Admin – Manage Blog – Add Blog

Admin – Manage Blog – Edit Blog

8.4 User Tests

What do you think about managing users?

Easy to use.

Do you understand validation errors when entering incorrect data?

◉ Yes

◯ No

tps://docs.google.com/forms/d/1KxILH2IIDiCuiknSiFKwL9j-AxLAibx2zgUHa0t98Zs/edit#responses

'7/2019   SOFT352 - User Tests

What do you think of the registration process?

No confirmation E-mail?