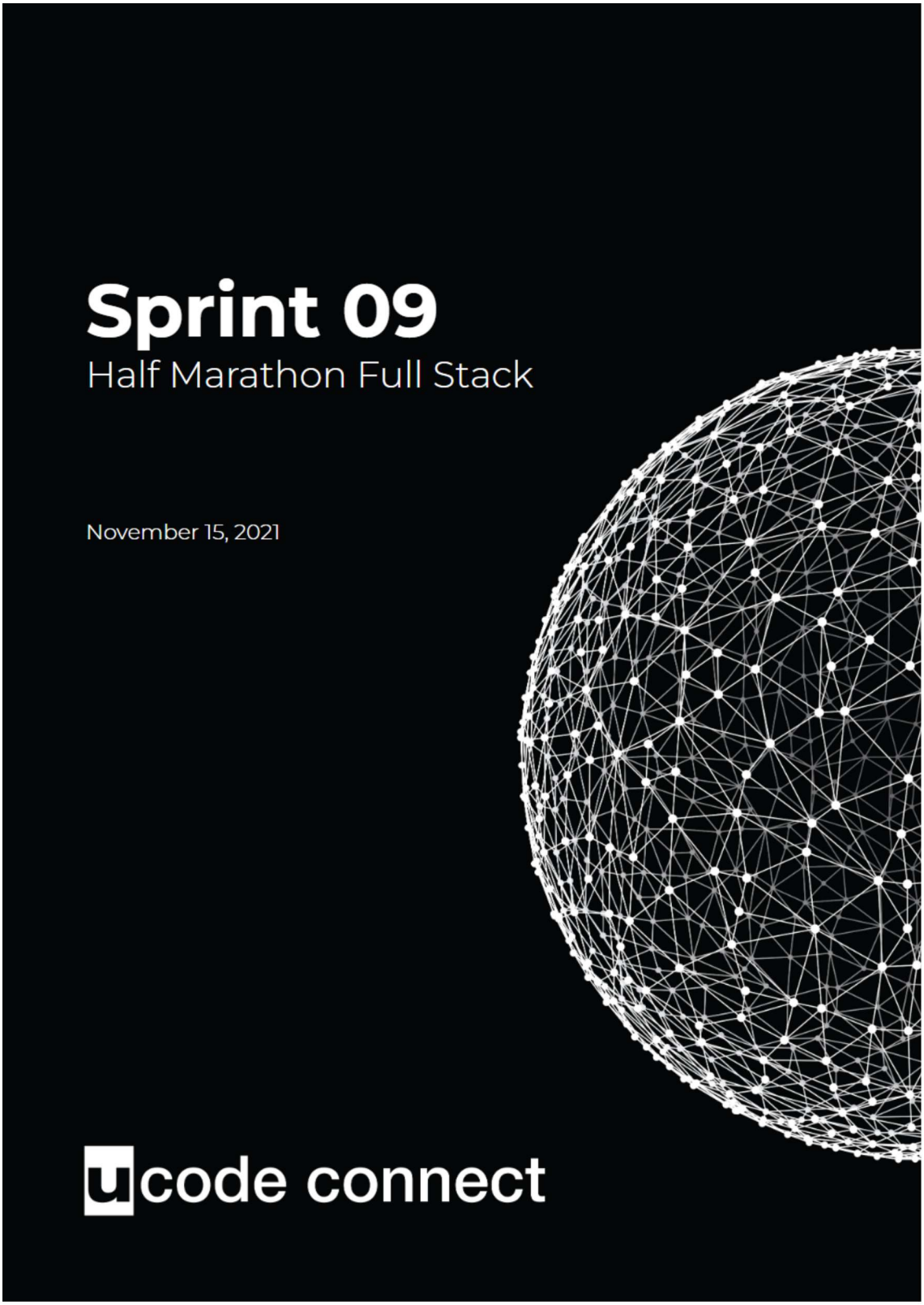
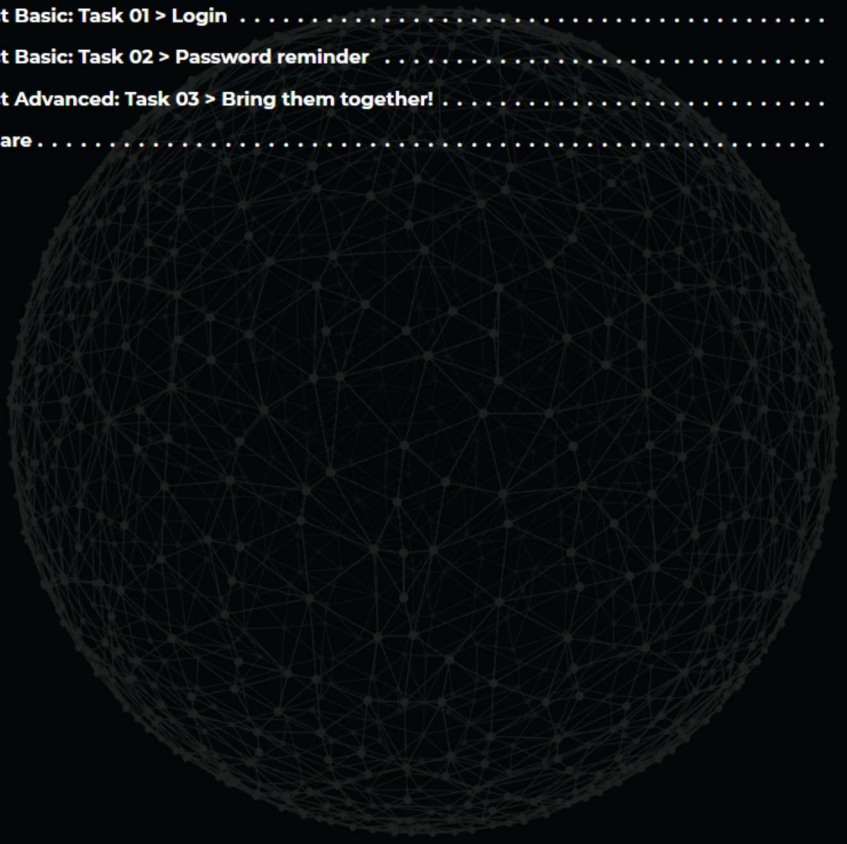

Sprint 09
Half Marathon Full Stack
November 15, 2021
ucode connect

Contents



ucode connect

Engage

DESCRIPTION

OH MY GOD!

Can you believe it?! It is the last Sprint in this Half Marathon!
It means that it's time to code like a strong and independent almost-junior developer that you are. Are you ready to meet real tasks? Do you want to know how to make a web service without the help of WordPress?

Today you will get acquainted with software design patterns in Node.js. They form the base of backend development. They can be considered best practices for solving common problems which can occur during designing an app or system.

The key aspect of most web services is user interaction. Among the most popular patterns for developing user interfaces is MVC. As it is impossible to be a good web developer without a basic understanding of how the internet works, we advise you to revise what you know about transmission of information packages and the role of the controller and router in this process.

Also, the knowledge of URL structure is obligatory in this Sprint, because you are about to parse the address bar to understand what the user of your web service wants to accomplish.

Make the last effort and you will get to the finish line!
Come on, you can definitely do it!

BIG IDEA

User's interaction with a web service.

ESSENTIAL QUESTION

How to handle user requests?

CHALLENGE

Build your first web service.

ucode connect

Sprint 09 | Half Marathon Full Stack **> 2**

Investigate

GUIDING QUESTIONS

We invite you to find answers to the following questions. By researching and answering them, you will gain the knowledge necessary to complete the challenge. To find solutions, ask the students around you and search the internet. We encourage you to ask as many questions as possible. Note down your findings and discuss them with your peers.

- What does MVC stand for and what does each component do?
- What are the advantages of using MVC?
- Which part of the MVC pattern does the user interact with?
- How is routing implemented in Node.js? Discover how to implement routing by using vanilla Node.js and then `Express framework`.
- What is the difference between static and dynamic routing?
- What is a URL? Describe the URL structure.
- What is a domain name?
- What is a `.htaccess` file? Why do we use it and where?
- What are the main error types in Node.js and how do they differ?
- How can you enable error reporting in Node.js?
- What is a 404 error?
- Which codes of server response do you know?

GUIDING ACTIVITIES

Complete the following activities. Don't forget that you have limited time to overcome the challenge. Use it wisely. Distribute tasks correctly.

- Discover how to submit a form without a submit button.
- Explore how to check the code of the page response.
- Find out what a perfect 404 page should look like and what it should contain.
- Get a basic understanding about URL parsing. Read official doc.
- Try to explain the MVC pattern to another student so that they understand it.
- Clone your git repository issued on the challenge page in the LMS.
- Brainstorm with other students and find more than one solution.
- Try to implement your thoughts in code.

ANALYSIS

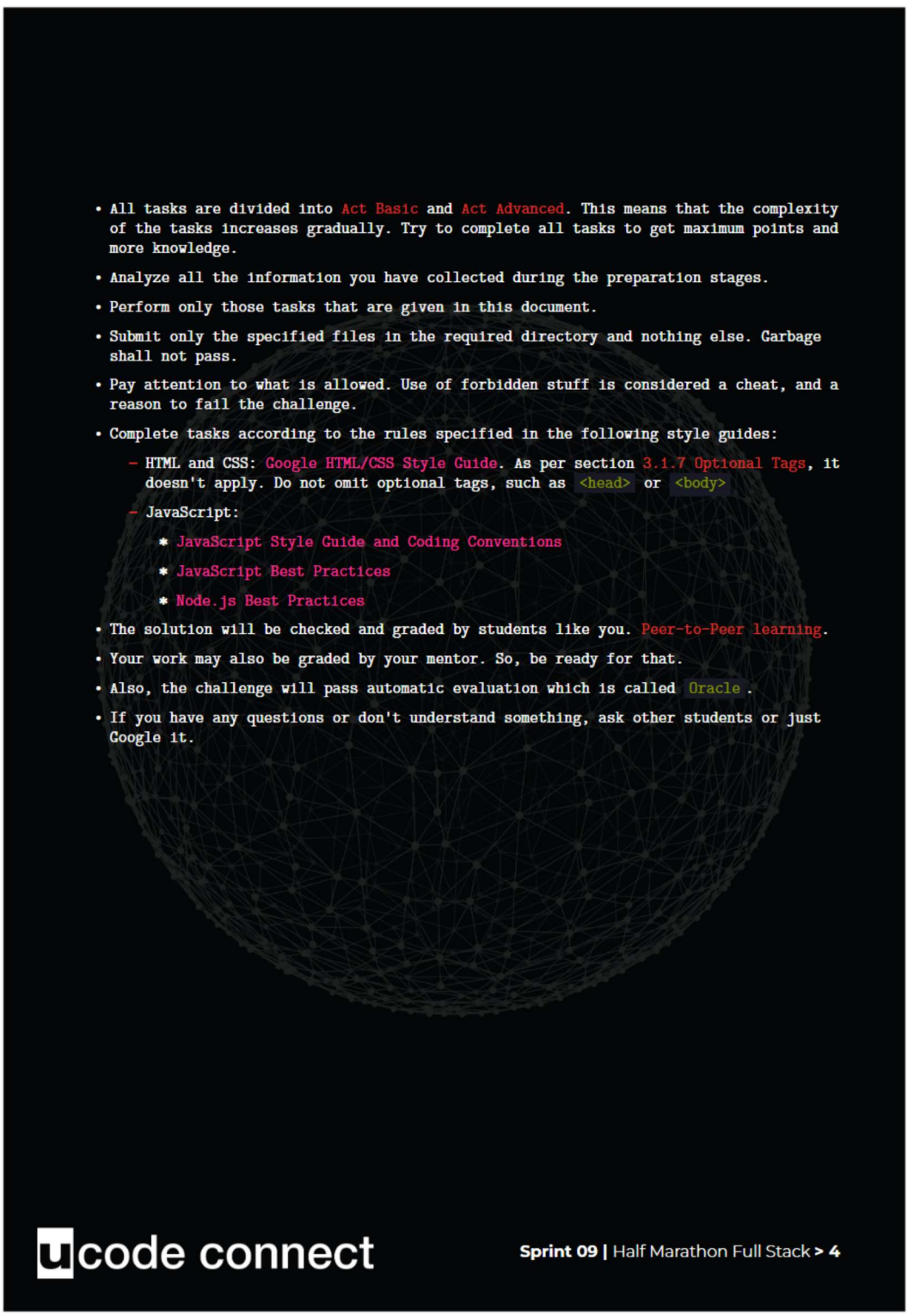
Analyze your findings. What conclusions have you made after completing guiding questions and activities? In addition to your thoughts and conclusions, here are some more analysis results.

- Be attentive to all statements of the story. Examine the given examples carefully. They may contain details that are not mentioned in the task.

ucode connect

Sprint 09 | Half Marathon Full Stack **> 3**

- All tasks are divided into Act Basic and Act Advanced. This means that the complexity of the tasks increases gradually. Try to complete all tasks to get maximum points and more knowledge.
- Analyze all the information you have collected during the preparation stages.
- Perform only those tasks that are given in this document.
- Submit only the specified files in the required directory and nothing else. Garbage shall not pass.
- Pay attention to what is allowed. Use of forbidden stuff is considered a cheat, and a reason to fail the challenge.
- Complete tasks according to the rules specified in the following style guides:
  - HTML and CSS: Google HTML/CSS Style Guide. As per section 3.1.7 Optional Tags, it doesn't apply. Do not omit optional tags, such as `<head>` or `<body>`
  - JavaScript:
    * JavaScript Style Guide and Coding Conventions
    * JavaScript Best Practices
    * Node.js Best Practices
- The solution will be checked and graded by students like you. Peer-to-Peer learning.
- Your work may also be graded by your mentor. So, be ready for that.
- Also, the challenge will pass automatic evaluation which is called Oracle .
- If you have any questions or don't understand something, ask other students or just Google it.



Act Basic: Task 00

NAME

Registration

DIRECTORY

`t00_registration/`

SUBMIT

`index.js, model.js, db.js, db.sql, config.json,`
`package.json, package-lock.json, models/user.js, public/*.css, public/*.html`

ALLOWED

HTML, CSS, JS, JSON, SQL

LEGEND

The global automatization process does not step over the S.H.I.E.L.D. A new innovation database named S.W.O.R.D. has been under development for the last three years, and now it's time to move forward. Director Fury needs to improve the registration of new agents in the system. Now they should create their profiles by themselves and not waste the time of the slave Director.

DESCRIPTION

Create a database db.sql with a table called users.
Create a simple form to register new users. The form must contain the following fields:

- login (must be unique)
- password
- confirm password (make sure to not let the form be submitted unless this field matches the password)
- full name
- email address (must be unique)

The fields (except *confirm password*) must exist in the database as well.

Handle errors correctly, show appropriate messages to a user if something went wrong. Certanly, show a success message if a user was created.

You may use the files from the previous Sprint (db.js, config.json, model.js, models/user.js).

Files that are to be served as is, such as html and css files, must be put into a public directory.

All your database manipulations must be written in the 'db.sql' file.

ucode connect

Sprint 09 | Half Marathon Full Stack **> 5**

Act Basic: Task 01

NAME

Login

DIRECTORY

`t01_login/`

SUBMIT

`index.js, model.js, db.js, db.sql, config.json,`
`package.json, package-lock.json, models/user.js, public/*.css, public/*.html`

ALLOWED

HTML, CSS, JS, JSON, SQL

LEGEND

The process of registration of new agents is going well, but the agent still can't understand what to do next. Do not let them share rumors but let them get access to their accounts.

DESCRIPTION

Create a simple form to allow login for registered users:

- the form must contain the following fields:
  - login
  - password
- a logged-in user must not see the login form, but must see:
  - the status (admin or user), which corresponds to the user's field in the database (create this field if you don't have it)
  - a button to log out and get back to the login page
- a corresponding message must appear after a successful or failed login
- all errors must be managed with this process

To test your login system use a database with users from a previous task.

ucode connect

Sprint 09 | Half Marathon Full Stack **> 6**

Act Basic: Task 02

NAME

Password reminder

DIRECTORY

`t02_password_reminder/`

SUBMIT

`index.js, model.js, db.js, db.sql, config.json,`
`package.json, package-lock.json, models/user.js, public/*.css, views/*.html`

ALLOWED

HTML, CSS, JS, JSON, SQL, nodemailer

LEGEND

Moving all agents to S.W.O.R.D. is going well. All units have access to the system. Recently, a new agent, Paulson, had missed his target and got a lot of injury from an enemy. The rehabilitation has not taken a lot of time, but the agent had partially lost his memory. Agent Paulson is back to the S.H.I.E.L.D. but cannot access the S.W.O.R.D. system. You need to do something about it.

DESCRIPTION

Create a simple form for the user to get a reminder about their password.
Only registered users can send a request to get a password reminder. You should get a password and send it to the user's email.

Remember! It's not secure to send passwords by email, so don't do it in real life!

SEE ALSO

nodemailer

ucode connect

Sprint 09 | Half Marathon Full Stack **> 7**

Act Advanced: Task 03

NAME

Bring them together!

DIRECTORY

`t03_bring_them_together/`

SUBMIT

`models/*.js, public/*.css, views/*.html, *.*`

ALLOWED

HTML, CSS, JS, JSON, nodemailer

LEGEND

Nick Fury is in a fury! S.W.O.R.D. had to become a great weapon to fight all unautomated processes, but for now it's just a bunch of useless stuff and no one can understand the purpose of each part.
We have some secret information from agent Paulson from Wakanda. It contains notes about how to join together all the stuff we've made before and can explain a lot of things. Use it to join all the parts and run the system.

DESCRIPTION

It's time to join everything you created today in one beautiful app. Use the MVC pattern for your architecture.
Compose all the files together and create a powerful app.
The main requirements are:

- the router must call the corresponding controller class depending on the URL passed in the address string
- the 404 page must be displayed if the controller does not exist
- store all your models in the `models/` folder
- each model must be named after the corresponding table in the database
- you must check if the user logged in before displaying a page for them
- if the user is not logged in, the user must see only the login page
- your controllers must be available to make the changes in the database if the user wants to do so

There are some required pages that you must implement:

- registration page
- login page
- password reminder page
- main page

ucode connect

Sprint 09 | Half Marathon Full Stack **> 8**

Share

PUBLISHING

Last but not least, the final stage of your work is to publish it. It allows you to share your challenges, solutions, and reflections with local and global audiences. During this stage, you will discover ways of getting external evaluation and feedback on your work. As a result, you will get the most out of the task and better understand your achievements and missteps.

To share your work, you can create:

- a text post, as a summary of your reflection
- charts, infographics or other ways to visualize your information
- a video, either of your work, or a reflection video
- an audio podcast. Record a story about your experience
- a photo report with a small post

Helpful tools:

- Canva - a good way to visualize your data
- QuickTime - an easy way to capture your screen, record video or audio (macOS)
- ScreenToGif - screen, webcam, and sketchboard recorder with an integrated editor (Windows)

Examples of ways to share your experience:

- Facebook - create and share a post that will inspire your friends
- YouTube - upload an exciting video
- GitHub - share and describe your solution
- Instagram - share photos and stories from ucode. Don't forget to tag us :)

Share what you've learned and accomplished with your local community and the world. Use #ucode and #CBLWorld on social media.