Web Application Development

Alexander Menshchikov

WAD 2020 Lecture Notes – #3 «Team work & authentication»

Team work & authentication

WAD 2020 Lecture Notes – #3 «Team work & authentication»

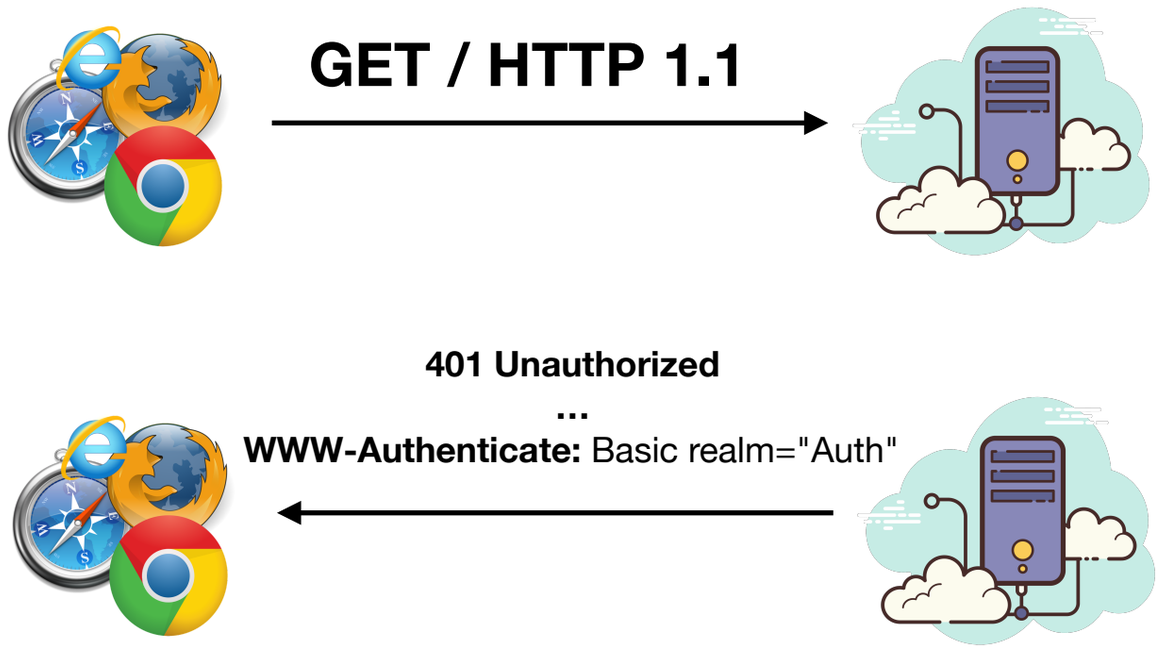
!?

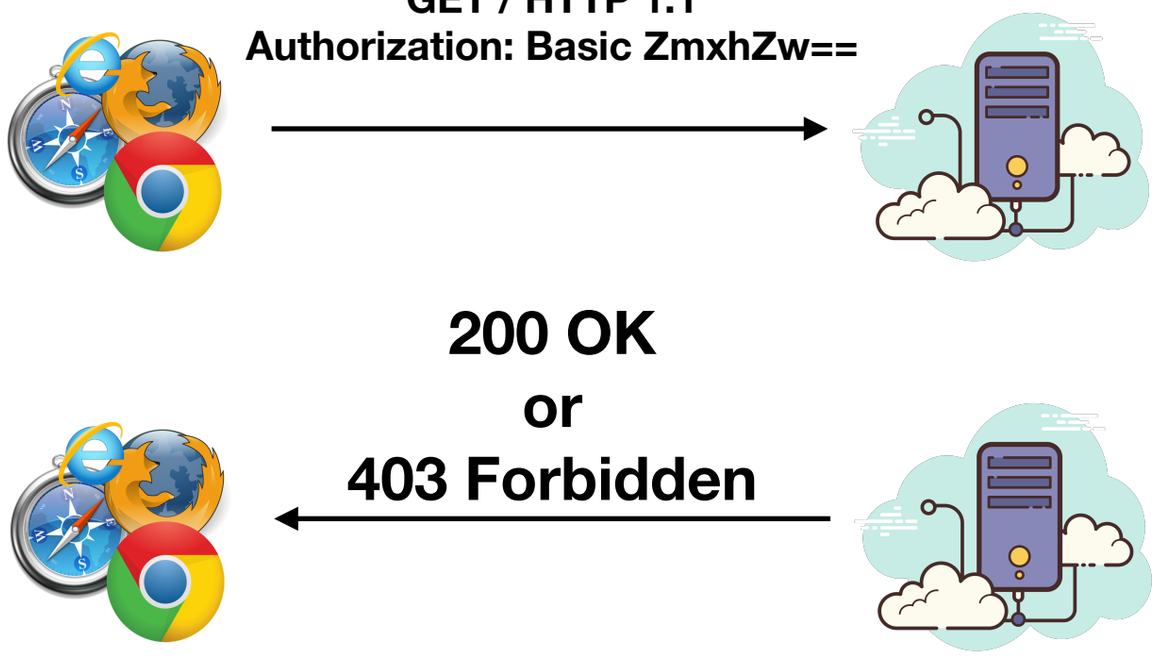
Quiz time

https://quiz.itmo.xyz/

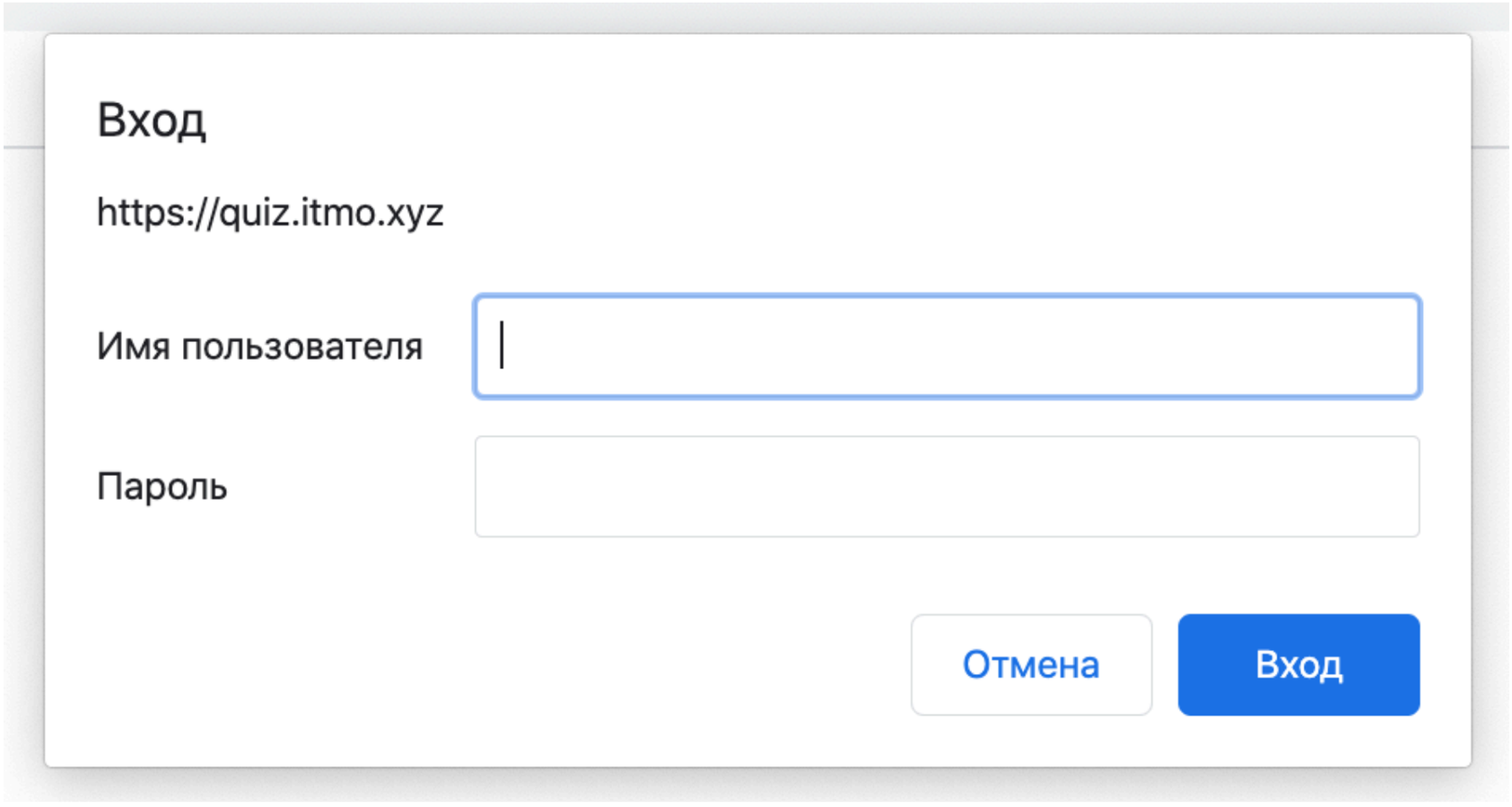
Authentication

Basic Authentication





Basic Authentication



Basic Authentication

👎 **Username and password in every request**

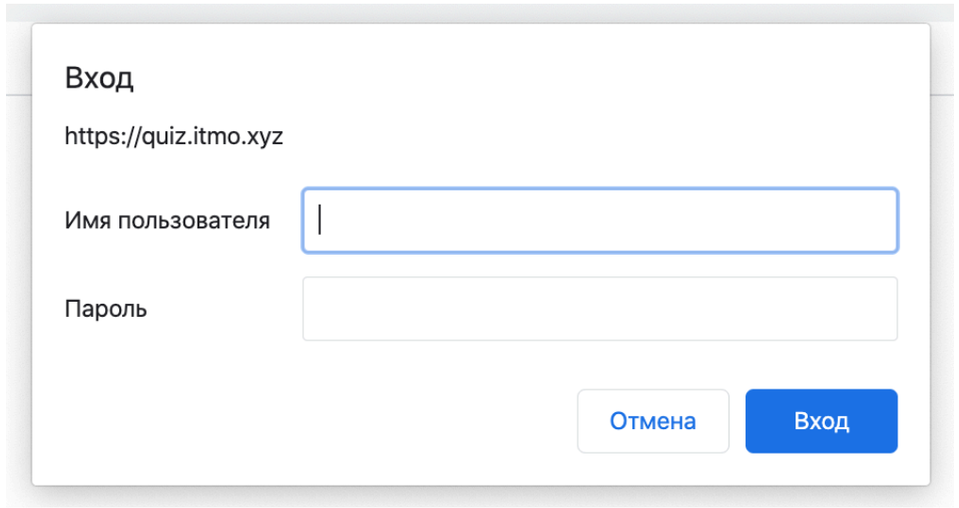
Less vulnerable under HTTPS

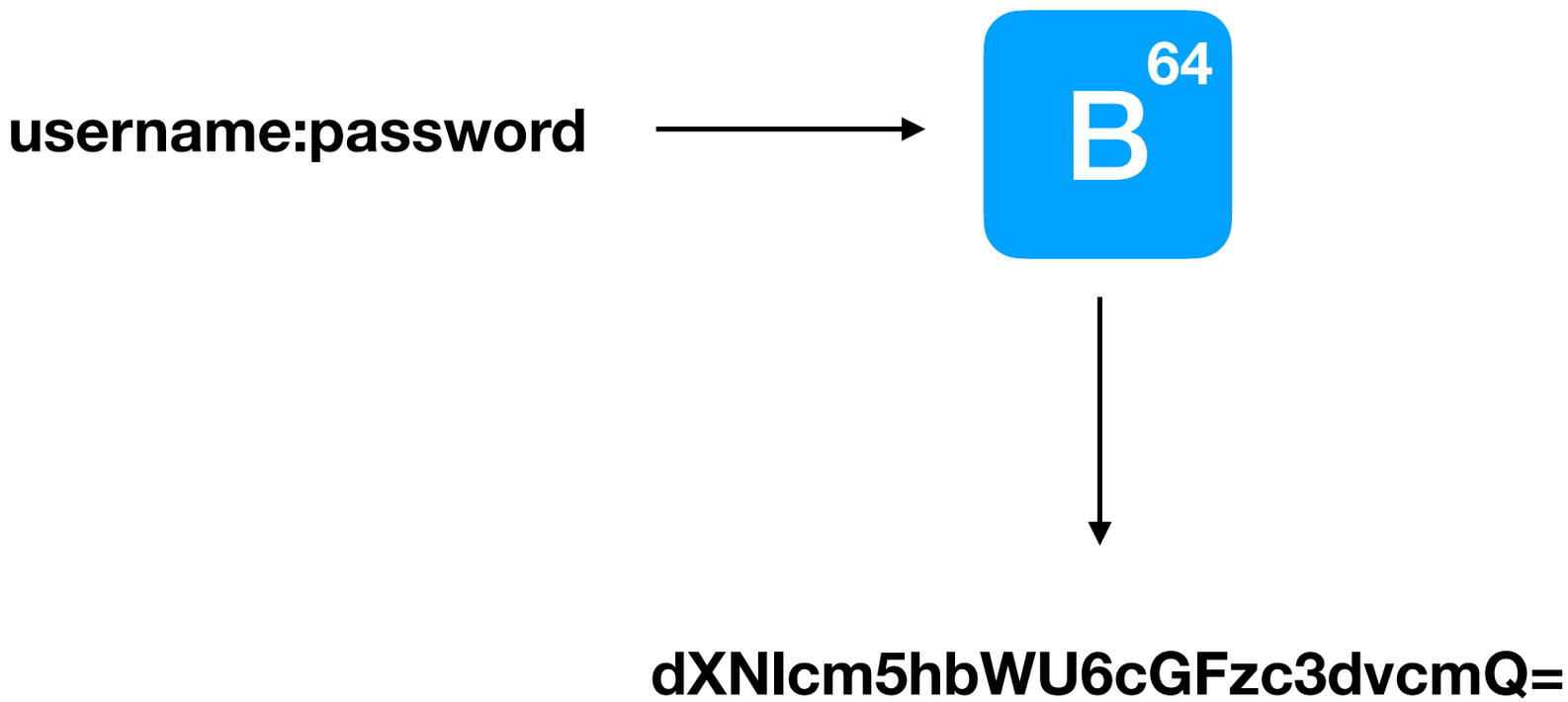
👎 **Password is the same and cannot be changed frequently**

👍 **Simple and fast**

👍 **Can be provided on the web-server side**

Base64 and basic auth





Cookie based auth

POST / HTTP 1.1
...
user=...&pass=...

200 OK
...
Set-Cookie: user=...&pass=...

GET / HTTP 1.1
...
Cookie: user=...&pass=...

200 OK
or
403 Forbidden

Cookie based auth

👎 **Username and password in every request**

Can be encrypted

👎 **Credentials can be extracted via JavaScript**

Can be protected

👍 **Still simple and fast**

👍 **Stateless**

Session based auth

POST / HTTP 1.1
...
user=...&pass=...

200 OK
...
Set-Cookie: session=...

GET / HTTP 1.1
...
Cookie: session=...

200 OK
or
403 Forbidden

Session based auth

👍 **No password transfer**

👎 **The same authentication token in every request**

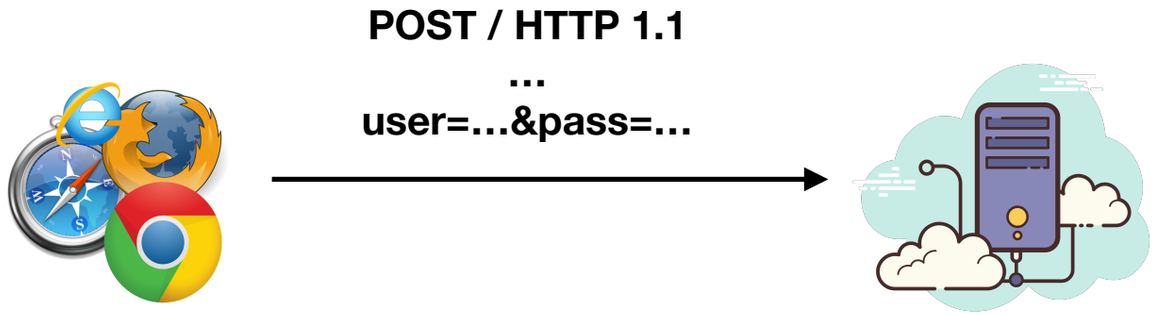
👎 **Token can be extracted via JavaScript**

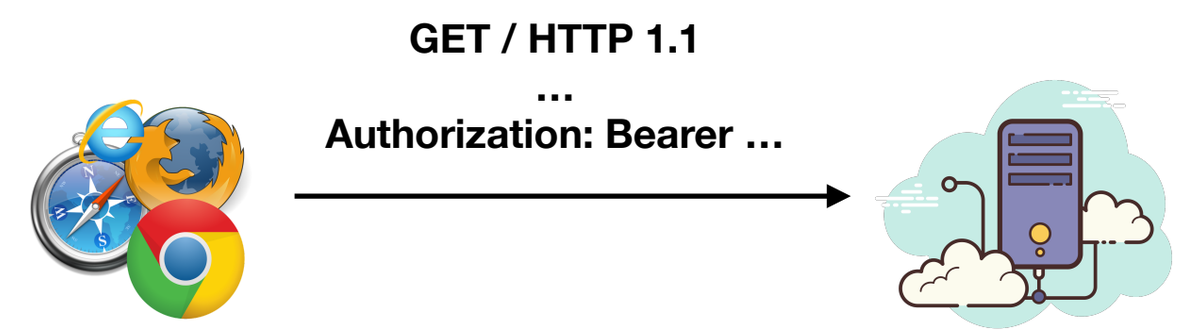
Can be protected

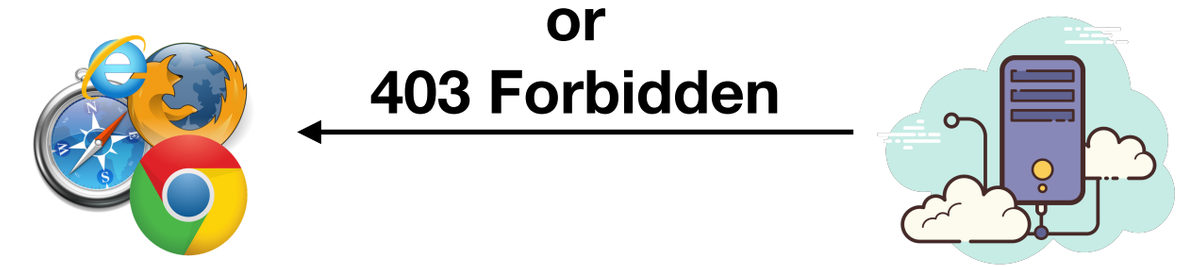
👍 **Simple**

Not stateless

Token based auth







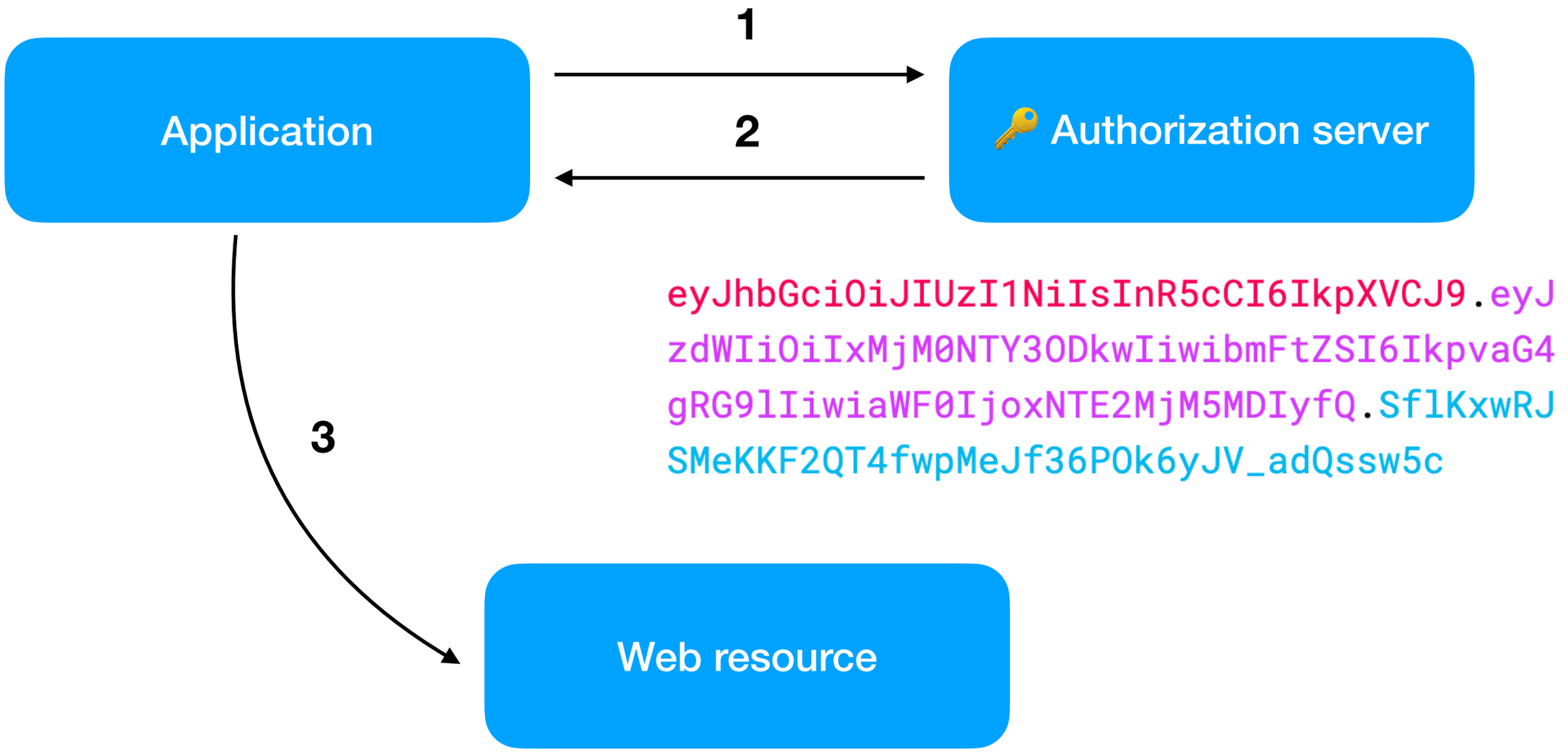
Token based auth

👍 **No password transfer**

👎 **More complicated**

👍 **Stateless**

JWT


1
2
Application
🔑 Authorization server
3
Web resource
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ
zdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4
gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJ
SMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c

Literature

- HTTP Authentication – https://developer.mozilla.org/en-US/docs/Web/HTTP/Authentication
- Base64 – https://en.wikipedia.org/wiki/Base64
- JWT – https://jwt.io/

Flask

Redirect

Use cases

- Expanding the reach
  - Synonyms (www.quiz.itmo.xyz –> quiz.itmo.xyz)
  - Typos (www.quis.itmo.xyz –> quiz.itmo.xyz)
- Moving to a new domain
  - ifmo.su –> codex.so
- Force HTTPS
  - http://wad.itmo.xyz –> https://wad.itmo.xyz

Redirect

GET / HTTP 1.1

301 Moved Permanently
Location: /new_url

GET /new_url HTTP 1.1

Permanent redirections

301 Moved Permanently — *GET*
308 Permanent redirect — *POST*

Temporary redirections

302 Found — *GET*
307 Temporary Redirect — *POST*

Special redirections

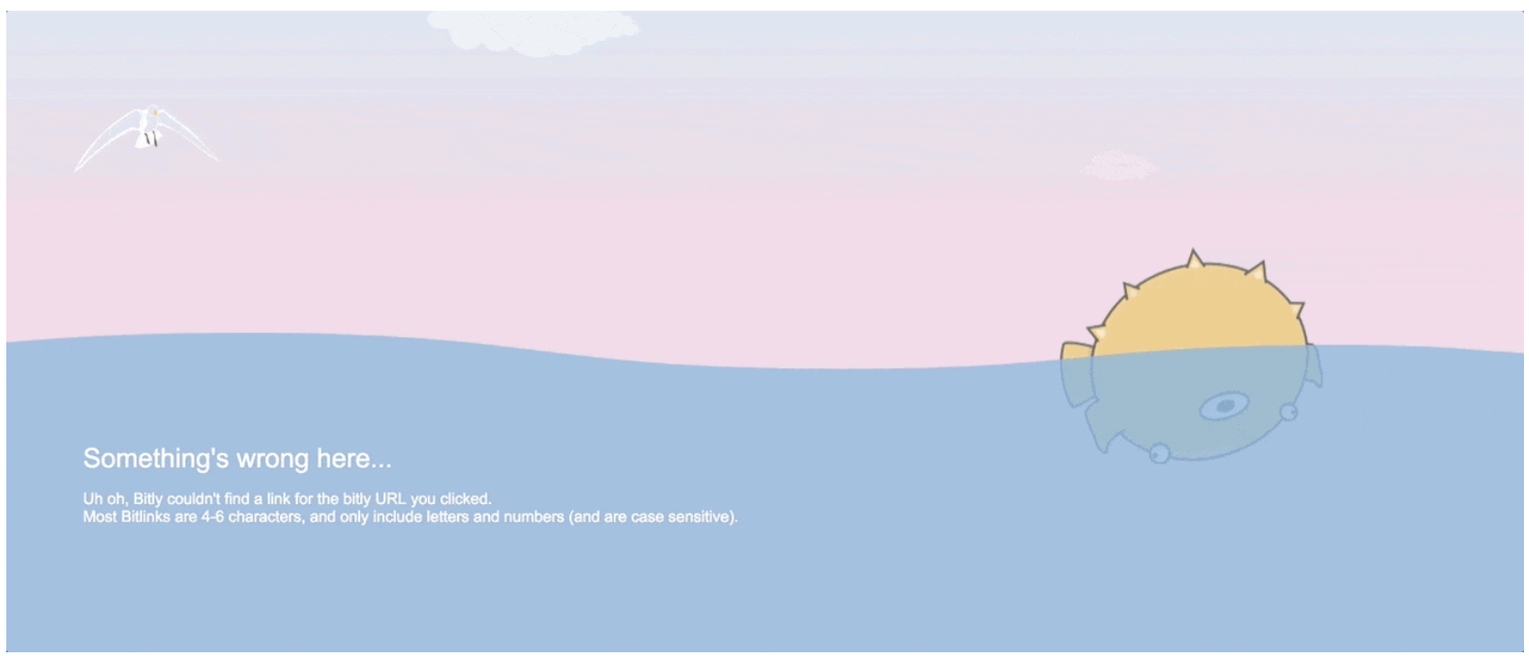
304 Not Modified

Errors

- Common error codes
  - 404 Not Found
  - 403 Forbidden
  - 500 Internal Server Error

```python
from flask import render_template

@app.errorhandler(404)
def page_not_found(e):
    # note that we set the 404 status explicitly
    return render_template('404.html'), 404
```



Flash

GET / HTTP 1.1

301 Moved Permanently
Location: /new_url
Cookie: message

```python
@app.route('/login', methods=['GET', 'POST'])
def login():
    error = None
    if request.method == 'POST':
        if request.form['username'] != 'admin' or \
                request.form['password'] != 'secret':
            error = 'Invalid credentials'
        else:
            flash('You were successfully logged in')
            return redirect(url_for('index'))
    return render_template('login.html', error=error)
```

GET /new_url HTTP 1.1

HTML with message

```jinja
<!doctype html>
<title>My Application</title>
{% with messages = get_flashed_messages() %}
  {% if messages %}
    <ul class=flashes>
    {% for message in messages %}
      <li>{{ message }}</li>
    {% endfor %}
    </ul>
  {% endif %}
{% endwith %}
{% block body %}{% endblock %}
```

Literature

- Redirect – https://flask.palletsprojects.com/en/1.1.x/quickstart/#redirects-and-errors

- Error handlers – https://flask.palletsprojects.com/en/1.1.x/patterns/errorpages/

- Flashing – https://flask.palletsprojects.com/en/1.1.x/patterns/flashing/

Demo

Team work

Team roles

- Project stakeholder
- Project manager
- Designer
- Frontend developer
- Backend developer
- Devops/admin
- Tester
- Content manager

Team roles example

- Project manager — Responsible for the project, write docs and slides
- Designer — Plan interfaces
- Frontend developer → HTML, CSS, JS
- Backend developer — Flask, databases
- Devops/admin — Deploy, security
- Testers — Test functionality

1 person can try several roles

Project roadmap

1. Project idea and schedule plan
2. Description of minimal functional product
3. Build minimal product in several iterations
4. Plan next features
5. Implement features
6. Prepare docs, deploy, test and present your project

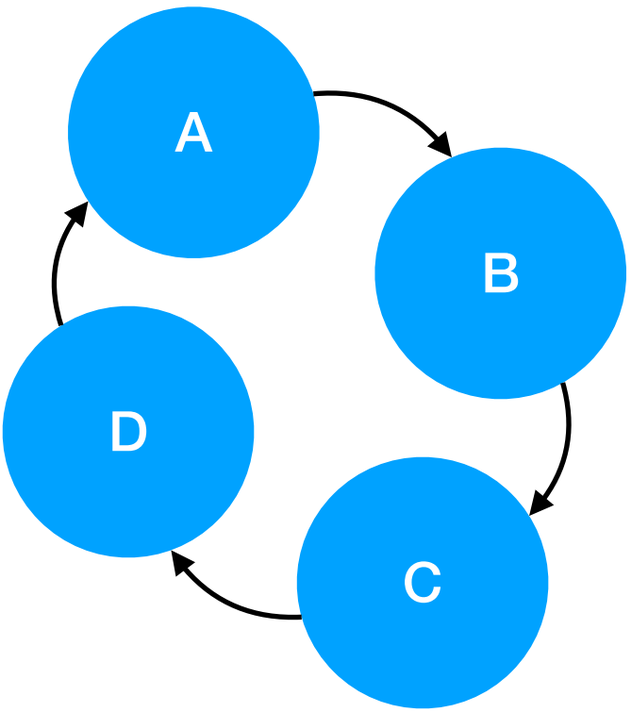
Communication

- Text
  - Telegram
  - Slack
- Audio and screen sharing
  - Zoom
  - Discord

Knowledge

- Google docs – https://drive.google.com/
- Notion – http://notion.so/
- Pinned messages in Telegram
- GitHub wiki – (like this) https://github.com/itmo-wad/WAD-Course-ITMO/wiki
- GitHub repository docs

Brainstorming



Make google docs template

Each person makes a copy

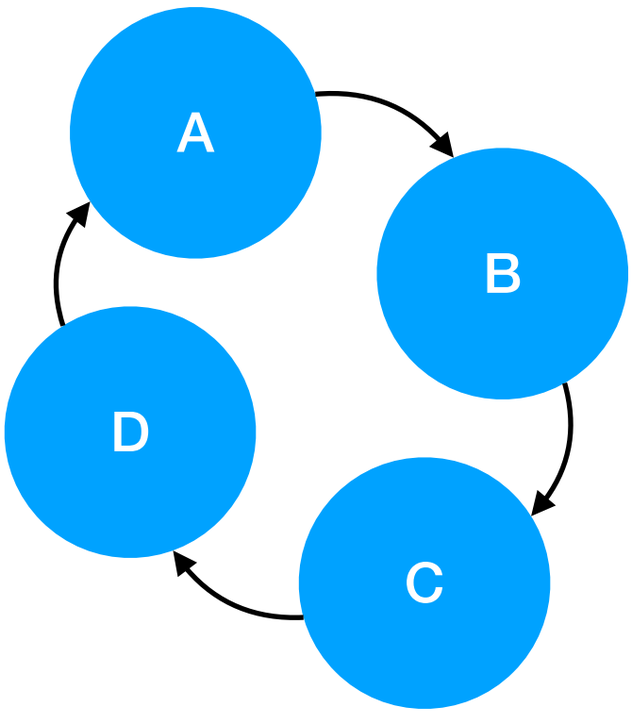
Write ideas in your doc for 30 minutes

Share link with edit access to the next person (A->B, B->C, …)

Add ideas to the new doc, append new details for the ideas

After 10 minutes share second document to C and take third from A …

Brainstorming



After 30 + 10 * 3 = 60 minutes

Take all 4 documents and merge ideas

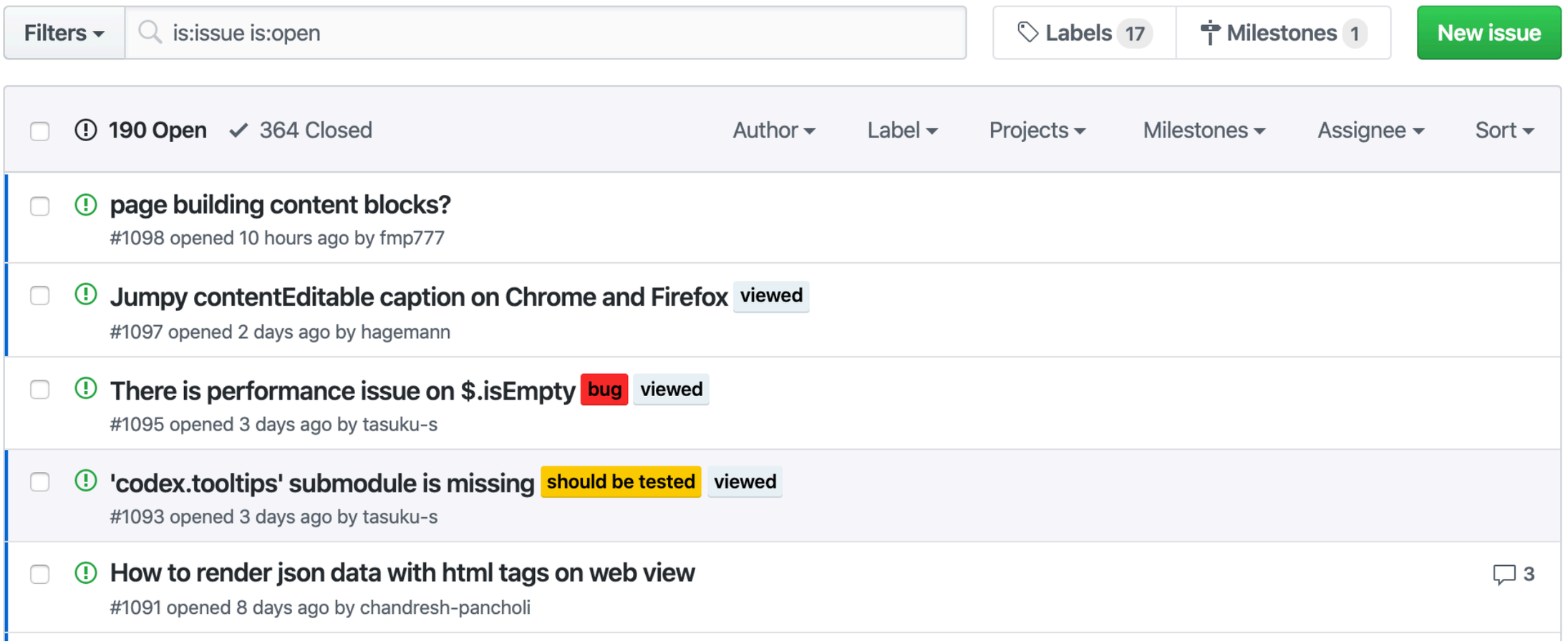
Now read carefully

And discuss in the voice chat

Best ideas worth realization

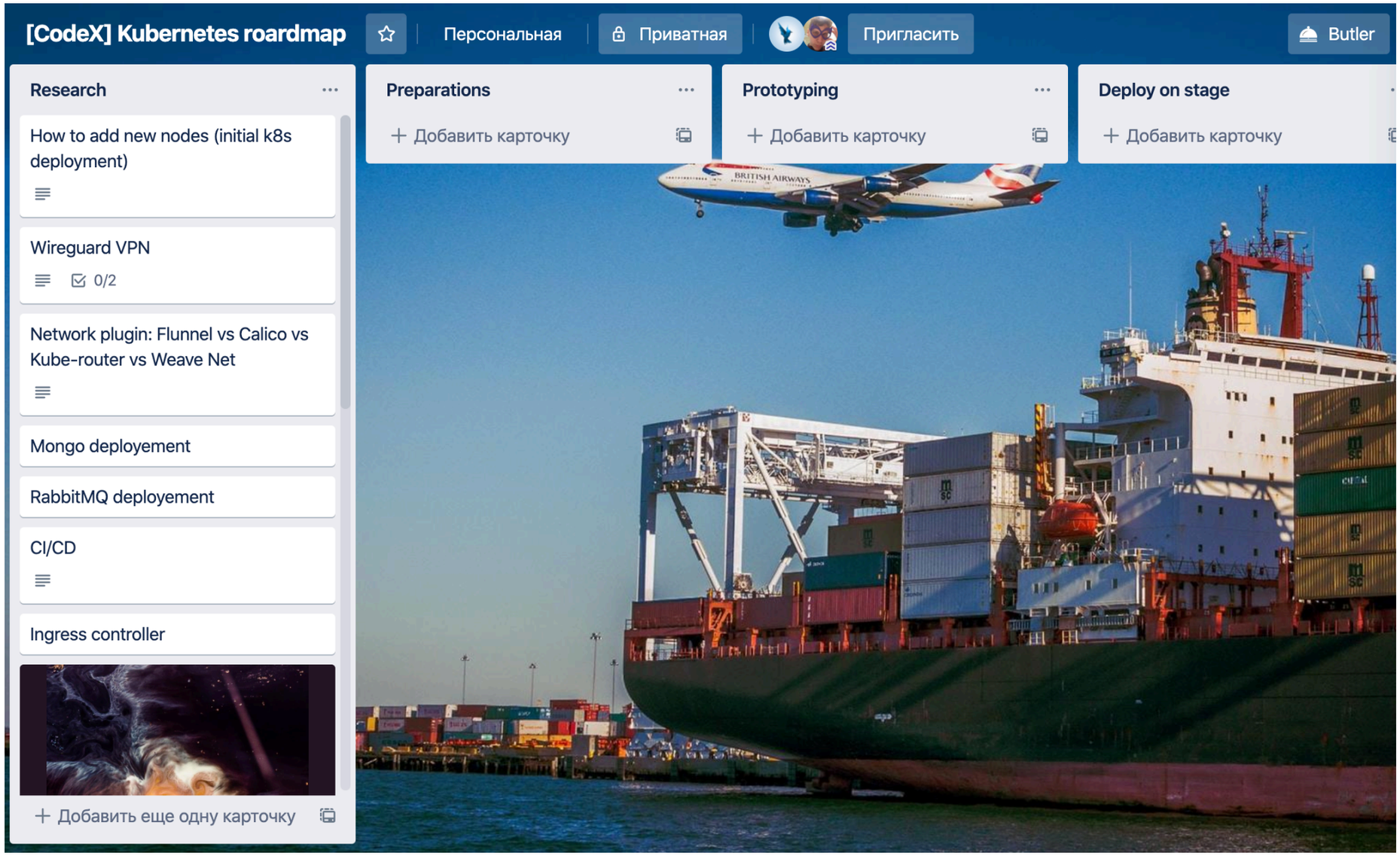
Task management

- GitHub issues



Task management

- Trello cards



Work process

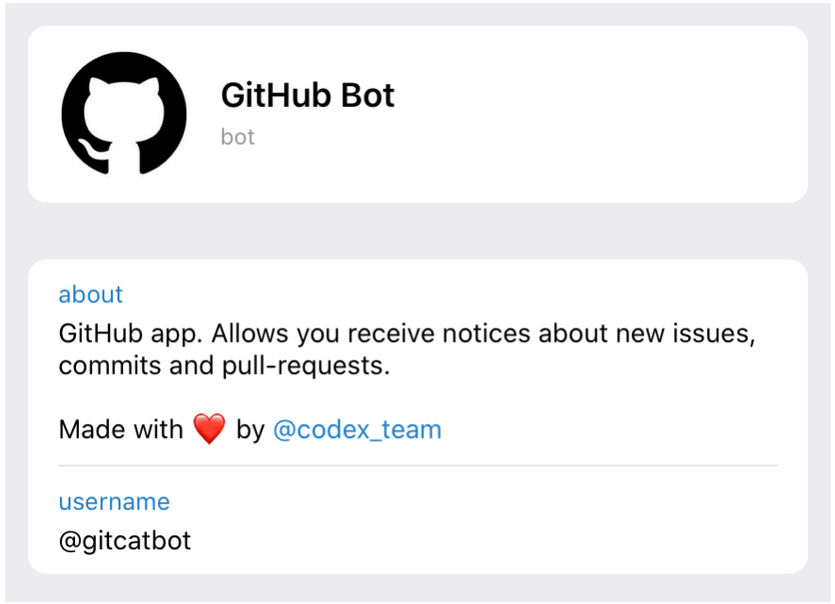
- Daily reports
- Weekly calls with task planning
- Be active in the chat

Automation

- Be informed about commits with the help of bots – https://github.com/codex-bot/github

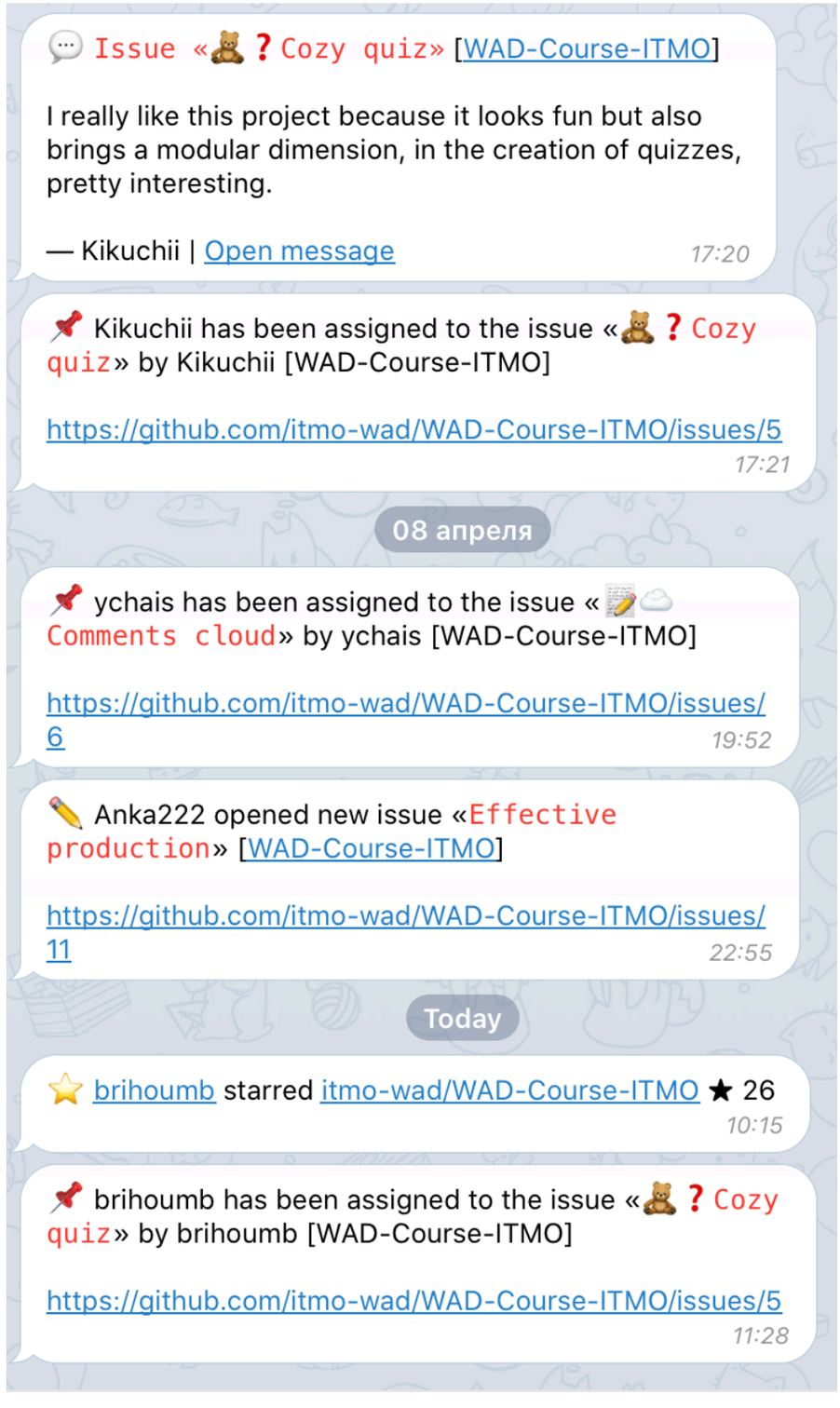
- Deploy free – https://dashboard.heroku.com/

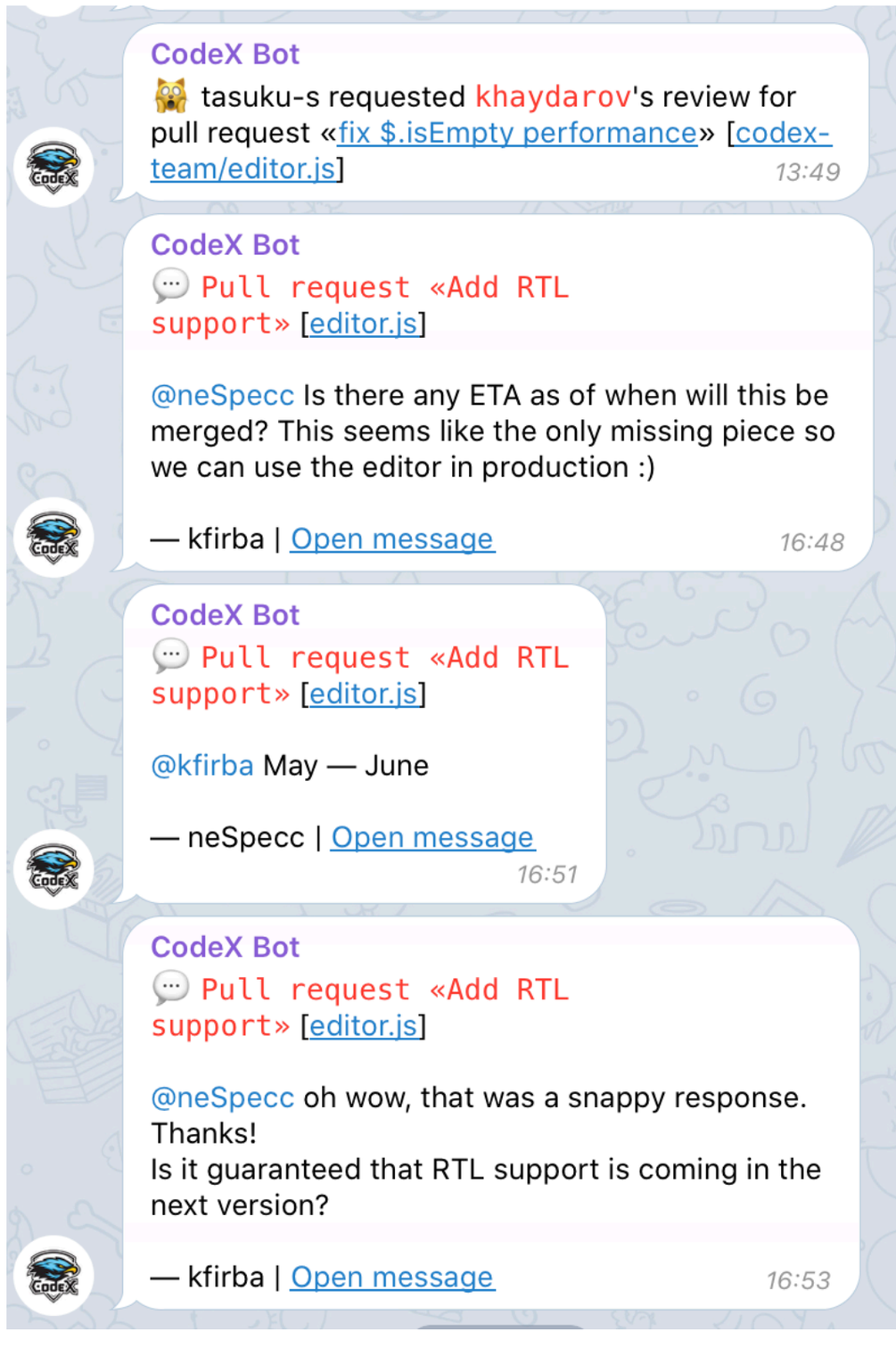
- Register and get free VPS for the first time – https://aws.amazon.com/free/?nc1=h_ls&all-free-tier.sort-by=item.additionalFields.SortRank&all-free-tier.sort-order=asc



@gitcatbot

Automation


💬 Issue «🧸❓Cozy quiz» [WAD-Course-ITMO]
I really like this project because it looks fun but also brings a modular dimension, in the creation of quizzes, pretty interesting.
— Kikuchii | Open message
17:20
📌 Kikuchii has been assigned to the issue «🧸❓Cozy quiz» by Kikuchii [WAD-Course-ITMO]
https://github.com/itmo-wad/WAD-Course-ITMO/issues/5
17:21
08 апреля
📌 ychais has been assigned to the issue «📝☁️Comments cloud» by ychais [WAD-Course-ITMO]
https://github.com/itmo-wad/WAD-Course-ITMO/issues/6
19:52
✏️ Anka222 opened new issue «Effective production» [WAD-Course-ITMO]
https://github.com/itmo-wad/WAD-Course-ITMO/issues/11
22:55
Today
⭐ brihoumb starred itmo-wad/WAD-Course-ITMO ★ 26
10:15
📌 brihoumb has been assigned to the issue «🧸❓Cozy quiz» by brihoumb [WAD-Course-ITMO]
https://github.com/itmo-wad/WAD-Course-ITMO/issues/5
11:28


CodeX Bot
🙀 tasuku-s requested khaydarov's review for pull request «fix $.isEmpty performance» [codex-team/editor.js]
13:49
CodeX Bot
💬 Pull request «Add RTL support» [editor.js]
@neSpecc Is there any ETA as of when will this be merged? This seems like the only missing piece so we can use the editor in production :)
— kfirba | Open message
16:48
CodeX Bot
💬 Pull request «Add RTL support» [editor.js]
@kfirba May — June
— neSpecc | Open message
16:51
CodeX Bot
💬 Pull request «Add RTL support» [editor.js]
@neSpecc oh wow, that was a snappy response. Thanks!
Is it guaranteed that RTL support is coming in the next version?
— kfirba | Open message
16:53

Assignment #3

Task #3 - Authentication

Reading → Coding → Deploy → Code-review

Reading

1. Flask Sessions: https://flask.palletsprojects.com/en/1.1.x/quickstart/#sessions
2. Flask cookies: https://flask.palletsprojects.com/en/1.1.x/quickstart/#cookies
3. Flask login: https://flask-login.readthedocs.io/en/latest/

Coding

Preparations

1. Install Python programming language **v3.8.2**: https://www.python.org/downloads/
2. Install Flask framework with **pip**: https://docs.python.org/3/installing/index.html#basic-usage

Sign up

Your Name

Your Email

Password

Repeat your password

I agree all statements in Terms of service

Register

Basic part

1. Create web application, which can authenticate user with password:
    - Listen on `localhost:5000`
    - Render authentication form on `http://localhost:5000/`
    - Return static images and files on `http://localhost:5000/static/<image_name>`
    - Has secret page for authenticated users on `http://localhost:5000/cabinet`
2. You are allowed to use any JS or CSS frameworks
3. You are allowed to use only Python programming language and Flask framework
4. You can hardcode pre-defined usernames and passwords inside your application

Cabinet

Store your passwords here

Logout

Optimal part

1. Add registration function to append new users on `http://localhost:5000/register/`
2. Allow users logout `http://localhost:5000/logout`
3. You can keep usernames/passwords in global storage

Challenging part

(Part for those, who already knows all that stuff)

1. Implement session storage based on files in `sessions` directory
2. Files should be encoded in **JSON** format
3. Session cookies should be signed with **HMAC** or any other digest method

Deploy

1. Register on GitHub: https://github.com/
2. Join our organization: https://github.com/itmo-wad/
3. You can create new personal repository or use repository for the previous tasks (but don't remove old files, just append new directories)
4. Commit and push your sources to GitHub. And don't forget to describe shortly what have you done in `README.md` file. Use Markdown format: https://guides.github.com/features/mastering-markdown/
5. *(optional)* Deploy you sources on Heroku or somewhere else and add link to the server to `README.md`

Code-review

1. Communicate in Telegram chat
2. Help others to complete the assignment

Group work task

1. Get approve from me
to your project idea till the next lecture
2. Describe your idea publicly on GitHub
3. Post link and short description to the chat

Practice time